VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY
DEPARTMENT OF CONTROL AND INSTRUMENTATION

DATOVÝ KONCENTRÁTOR
DATA CONCENTRATOR

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Petr Dvorský**
AUTHOR

VEDOUCÍ PRÁCE — **doc. Ing. Petr Fiedler, Ph.D.**
SUPERVISOR

BRNO 2021

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Kybernetika, automatizace a měření**

Ústav automatizace a měřicí techniky

Student: Bc. Petr Dvorský ***ID:*** 191395

Ročník: 2 ***Akademický rok:*** 2020/21

NÁZEV TÉMATU:

Datový koncentrátor

POKYNY PRO VYPRACOVÁNÍ:

Cílem diplomové práce je vytvořit univerzální datový koncentrátor pro záznam dat z různých zdrojů. Součástí diplomové práce je implementace logování dat do interního úložiště a synchronizace na cloud.

1. Seznamte se s dostupnými automatickými měřicími systémy.
2. Definujte požadavky na datový koncentrátor pro měřicí systém.
3. Navrhněte koncepci zařízení a HW.
4. Realizujte HW.
5. Navrhněte SW.
6. Implementujte SW.
7. Otestujte zařízení a diskutujte dosažené výsledky.

DOPORUČENÁ LITERATURA:

Katalogové listy k zvoleným součástkám.

Termín zadání: 8.2.2021 ***Termín odevzdání:*** 17.5.2021

Vedoucí práce: doc. Ing. Petr Fiedler, Ph.D.

doc. Ing. Petr Fiedler, Ph.D.
předseda rady studijního programu

ABSTRAKT

Tato diplomová práce se zabývá návrhem a realizací modulárního Datového koncentrátoru pro nejrůznější typy měření v různorodých podmínkách. V práci je popsána koncepce zařízení, dále jeho základní principy, postup jeho návrhu a jeho funkčnost. Taktéž je zde popsán navržený lightweight protokol pro komunikaci Datového koncentrátoru se snímači, standardizovaný XML formát pro ukládání měřených dat, či využití cloudového prostředí v rámci měření. Elektrický návrh zařízení stejně jako návrh desky plošných spojů byl vytvořen v rámci softwaru Eagle Autodesk (Eagle Autodesk EDA). Řídicí software pro použitý mikrokontrolér ESP32-WROOM-32 je založen na operačním systému reálného času FreeRTOS a vytvořený v rámci frameworku ESP-IDF v C/C++. Jako vývojové prostředí pro toto programové vybavení bylo zvoleno Visual Studio Code s rozšíření PlatformIO. Pro Datový koncentrátor vybranou a využitou cloudovou platformou je ThingSpeak od společnosti MathWorks, která využívá některé prvky z prostředí Matlab.

KLÍČOVÁ SLOVA

Data koncentrátor, Eagle, DPS, ESP32-WROOM-32, nRF24L01+, DS3231, SIM808, MicroSD, Visual Studio Code, PlatformIO, ESP-IDF, C/C++, FreeRTOS, XML, Lightweight protokol, Cloud, ThingSpeak, Matlab

ABSTRACT

The topic of this thesis is the design and realization of a modular Data concentrator for various types of measurements in diverse conditions. The device conception, basic principles, design and functionality are described herein. Also, the lightweight protocol design for radio communication, standardized XML format for data storage and cloud usage are described. The electrical design of the device as well as the design of the printed circuit board was made using the Eagle Autodesk electronic design automation software (Eagle Autodesk EDA). The control software written in C/C++ for a target microcontroller (ESP32-WROOM-32) is based on a FreeRTOS platform and ESP-IDF framework. An IDE for managing this software is Visual Studio Code with PlatformIO extension. Selected and used Cloud Platform is ThingSpeak from Mathworks, which uses certain components from Matlab platform.

KEYWORDS

Data concentrator, Eagle, PCB, ESP32-WROOM-32, nRF24L01+, DS3231, SIM808, MicroSD, Visual Studio Code, PlatformIO, ESP-IDF, C/C++, FreeRTOS, XML, Lightweight protocol, Cloud, ThingSpeak, Matlab

DVORSKÝ, Petr. *Datový koncentrátor*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky, 2021, 117 s. Diplomová práce. Vedoucí práce: doc. Ing. Petr Fieldler, Ph.D.

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Bc. Petr Dvorský
VUT ID autora:	191395
Typ práce:	Diplomová práce
Akademický rok:	2020/21
Téma závěrečné práce:	Datový koncentrátor

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu doc. Ing. Petru Fiedlerovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci. Dále taktéž Ing. Ondřeji Baštánovi, Ing. Tomáši Benešlovi a v neposlední řadě Ing. Jakubu Streitovi.

Obsah

Seznam obrázků

Seznam tabulek

Seznam výpisů

Úvod

Tato diplomová práce vznikla za účelem vytvoření modulárního zařízení, které bude schopno koncentrovat a uchovávat nejrůznější naměřená data a dále je vhodně zpracovávat a distribuovat do nadřazeného systému.

Práce se tedy zabývá vytvořením Datové koncentrátoru, který má za úkol komunikovat s jednotlivými měřicími zařízeními, senzorickými jednotkami, se stejným komunikačním rozhraním a koncentrovat v sobě jimi naměřená data s dalšími možnostmi exportu do nadřazených systémů.

Vytvoření takovéhoto zařízení zahrnuje zprvu principiální a ideový návrh (Kapitola 1), na základě kterého je realizován výběr elektrotechnických součástek a modulů, kompletní elektrické schéma Datového koncentrátoru, realizace desky plošných spojů a její ruční osazení (Kapitola 2).

V neposlední řadě je v práci pojednáno o programovém vybavení takovéhoto zařízení pro odzkoušení jeho základní funkčnosti výpočetního jádra a k němu přidružených periferií a dále dodání kompletního programového vybavení zaručující všechny vytyčené cíle v rámci měření, komunikace, ukládání dat a exportu do nadřazeného systému (Kapitola 3).

Dále práce diskutuje výběr, zprovoznění a použití nadřazeného systému, který je v tomto případě zastoupen cloudovou platformou ThingSpeak (Kapitola 4).

Práce se taktéž v kapitole 5 zamýšlí nad nejrůznějšími problémy, které vyvstaly v průběhu návrhu a realizace takovéhoto zařízení a jejich řešení. Dále zde práce diskutuje testovací provoz zařízení a dosažené výsledky.

Všechna odborná terminologie, stejně jako zkratky a symboly nacházející dále v práci, jsou vysvětleny přímo v textu a taktéž uvedena na konci dokumentu v seznamu Symbolů, veličin a zkratek. (str. 90)

Tato práce navazuje textem i odborností na mou semestrální práci [1], která se věnovala totožné problematice a byla zaměřené na stejné téma.

1 Teoretická část semestrální práce

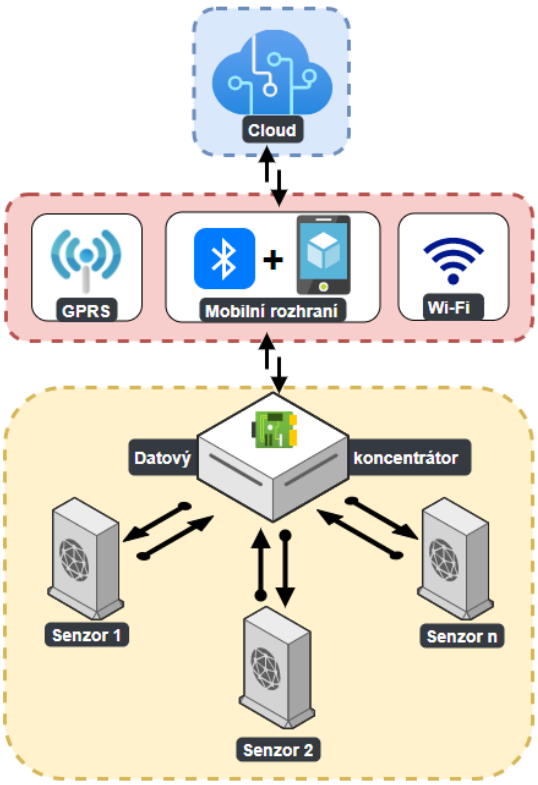
Tato kapitola se zabývá rozborem koncepce a využití samotného Datového koncentrátoru, následně vydefinováním požadavků na takovýto koncentrátor, rešerší vhodného použitého softwaru pro dokumentaci, realizaci a zprovoznění Data koncentrátoru a v neposlední řadě vhodného výrobce desek plošných spojů pro takovéto zařízení.

1.1 Koncepce Datového koncentrátoru

Hlavním důvodem pro vytváření dále popisovaného Data koncentrátoru je reakce na ne zcela nový požadavek vyvstávající v technických, ale i v různých dalších odvětvích, a to mít jednoduchou možnost měřit, ukládat a dále exportovat data v terénních a jiných neobvyklých podmínkách. Za tím účelem je třeba mít k dispozici jednoduché přenosné modulární zařízení, které dokáže data s pomocnými senzorickými moduly určenými přímo pro danou měřenou aplikaci přes jednotné komunikační rozhraní komunikovat a dále měřené data z těchto modulů extrahovat a schraňovat. Koncentrátor by měl takto získaná surová data nejen nejen schraňovat na lokálním úložišti, avšak dále by měl extrahovaný obsah nahrávat na Cloud, kde lze data dále zpracovat, uvést je do širšího kontextu a vdechnout jim tak hlubší smysl. K takto zpracovaným datům následně poskytnout v reálném čase vhodné vizuální vyjádření.

Jak již bylo naznačeno výše, takovýto koncentrátor by měl nalézt využití primárně v mobilních aplikacích, kdy je úloha měření zkomplikována nutným pohybem objektu a standardní aparaturu tedy nelze použít vůbec nebo jen s velkými obtížemi. Často se taktéž jedná o aplikace, kde není kladen až takový důraz na přesnost a kvalitu měření, ale na přibližné změření parametrů a jejich celkový, často fyzikální, kontext. Dalším možných polem užití tohoto koncentrátoru je měření systému s rozlehlými/rozprostřenými parametry, tedy měření, která jsou mnohabodová a prostorově náročná.

Aby takovéto možnosti mohly být v rámci koncentrátoru obsaženy, je zcela nutné, aby koncentrátor poskytoval dostatečnou modularitu v rámci možností měřit nejrůznější parametry a přitom měl jednotná rozhraní pro komunikaci s dalšími jeho funkčnost kompletujícími moduly. Modularita v tomto případě bude poskytnuta v možnosti připojení jakékoliv senzorické jednotky se společně definovaným rozhraním. Toto rozhraní, které bude následně podrobněji diskutováno později, by z povahy věci mělo být bezdrátové pro korektní splnění výše kladených požadavků na takovýto koncentrátor. Dále pro komunikaci s nadřazenými systémy by měl koncentrátor poskytovat vícero rozhraní, tedy možností, jak z něj koncentrovaná data exportovat.



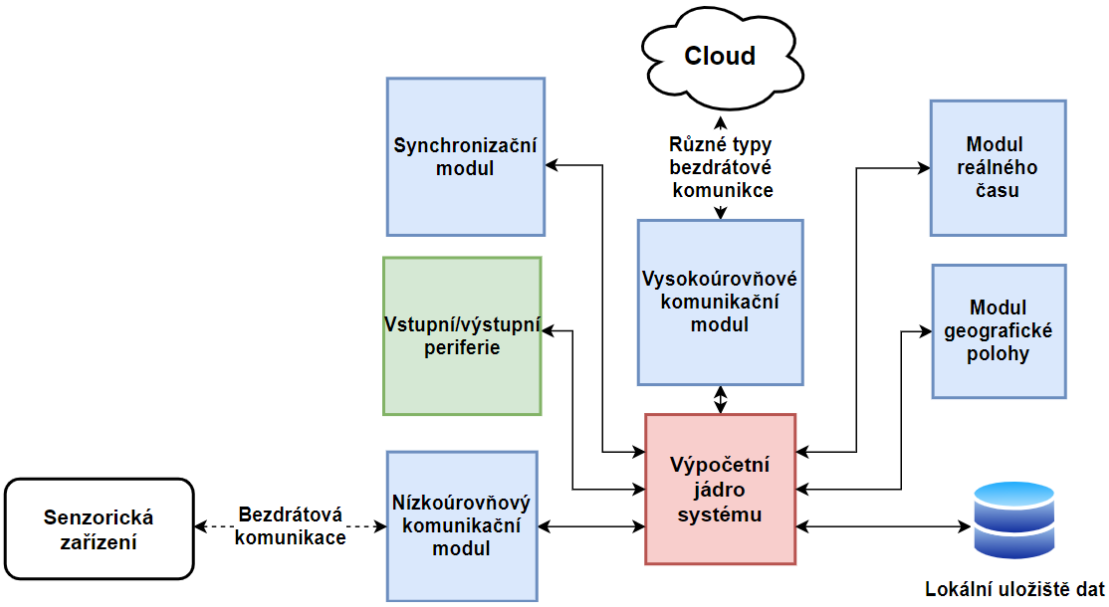
Obr. 1.1: Koncepce Datového koncentrátoru

Pro zasazení tohoto zařízení do kontextu a lepší vizualizaci řešeného problému a funkce samotného zařízení je zde obrázek 1.1. Ten vyobrazuje konkretizovanou koncepci takovéhoto zařízení, jeho principiální funkce a hierarchii. Jednotlivé prvky, jejich realizace a kompletní rozbor je však uveden v následujícím textu.

1.2 Obecný princip zařízení

Níže vyobrazené schéma (Obr. 1.2) popisuje základní návrh zařízení Data koncentrátoru. Aby koncentrátor obsáhl všechny požadované vlastnosti, které jsou na něj kladeny, musí disponovat mnoha periferními moduly, jež tyto funkcionality nabízejí, a dále výpočetním jádrem, které bude jednotlivé moduly sdružovat a vhodně povelovat.

Jako prvního si lze všimnout nízkoúrovňového komunikačního modulu. Jak vyplývá z výše zmíněného, pro komunikaci s jednotlivými senzorickými jednotkami bude zapotřebí použít takový modul, který dokáže obousměrně komunikovat s více zařízeními. Na této úrovni se budou komunikovat jednotlivá surová data, obsahující



Obr. 1.2: Principiální blokové schéma Data koncentrátoru

binární, či analogové hodnoty s časovou značkou a dalšími jen nejnutnějšími arbitrážními daty. Z tohoto modelového případu lze tedy usoudit, že se jedná o komunikaci jednoduchou, a tedy nízkoúrovňovou.

Dále si lze všimnout modulu reálného času a geografického modulu. Tyto moduly jsou zde potřebné z důvodu kontextualizace surových dat vzhledem k času a prostoru. Modul reálného času bude poskytovat aktuální přesný čas pro dané zařízení, geografický modul následně poskytne aktuální přesnou polohu v podobě GNSS souřadnic (primárně pro popis aktuální zeměpisné šířky, délky a nadmořské výšky daného místa).

Následným důležitým aspektem navazujícím na výše zmíněné moduly je synchronizační modul Datového koncentrátoru. Význam tohoto modulu tkví v globální synchronizaci právě aktuálního reálného času na koncentrátoru s jednotlivými, k němu připojenými, senzorickými jednotkami. Toto je především důležité z důvodů latencí a jiných prodlev v rámci nízkoúrovňové komunikace dat. Pakliže bychom jednotlivé senzorické jednotky nesynchronizovali, docházelo by k situaci, kdy by měřeným datům byla časová značka přidělena až v rámci Datového koncentrátoru. Samozřejmě je možné namítnout, že o problematice této skutečnosti rozhoduje přímo daná aplikace, tedy jaké máme nároky na přesnost samotných časových značek, ovšem je třeba počítat s "worst case scenario" a z toho důvodu požadujeme časovou značku co nejpřesnější vzhledem k danému měřicímu bodu a jeho naměřené hodnotě. V takovémto případě je jediným korektním řešením přiřazovat časovou značku měření

přímo v rámci měření senzorické jednotky. Synchronizační modul tedy zajistí, že všechny senzorické jednotky budou mít pojem o aktuálním čase a tím pádem bude čas mezi všemi zařízeními homogenní.

Na synchronizační modul navazují vstupní a výstupní periferie. Takovéto periferie jsou zde z důvodu uživatelského rozhraní umožňujícího využívat základní vstupy a výstupy z Datového koncentrátoru, ať již ve vizuální nebo mechanické, či elektrické podobě. Bude zde tedy vytvořeno takové rozhraní, které by umožňovalo oživení a inspekci zařízení jak na úrovni hardwarové tak i softwarové a dále také rozhraní pro ovládání některých funkcionalit Datového koncentrátoru.

Koncentrátorem přijatá data ze senzorických jednotek musí být dále vhodně zpracována a uložena. K tomu má sloužit modul lokálního úložiště dat. Zde budou všechna data vhodně označena, hierarchicky uspořádána a uložena nejlépe ve formátu jednoho ze standardně používaných značkovacích jazyků jako např. XML, či JSON. Takovýmto postupem docílíme toho, že při exportu dat z lokálního úložiště do nadřazeného systému budou data dále nejen jednoduše zpracovatelná, ale taktéž čitelná a srozumitelná pro samotného uživatele.

Posledním vyobrazeným modulem je vysokoúrovňový komunikační modul. Tento modul má za úkol zprostředkovávat komunikaci mezi Datovým koncentrátorem a vyššími nadřazenými systémy. V tomto případě se může jednat přímo o export nashromážděných dat do cloudového prostředí, avšak nemusí to být nutností. Data mohou být taktéž exportována na jiná zařízení vyšší úrovně (mobilní telefon, osobní počítač), která další zpracování a případný export dat na do cloudového prostředí obstarají samy.

Poslední a prakticky nejdůležitější částí systému je jeho samotné jádro. To má za úkol vhodně využívat prostředků poskytovanými výše zmíněnými moduly, řízení jednotlivých použitých modulů, zprostředkování vnitřní komunikaci mezi těmito jednotlivými moduly a v neposlední řadě vhodně využívat prostředků poskytovanými těmito moduly. Samotné jádro by mělo současně zajišťovat v souběhu s výše zmíněnými funkcionalitami i správu základních funkcionalit, jako je například správa napájení, chybových stavů a veškerých plánovaných i neobvyklých událostí.

Všechny takto navržené moduly a výpočetní jádro budou na reálném zařízení implementovány jako kombinace hardwarových a softwarových prostředků. Jejich přesná podoba a implementace bude však podrobněji přiblížena v následujících kapitolách.

1.3 Porovnání dostupných automatických měřicích systémů

V rámci práce vyvíjené zařízení je svým zaměřením na rozhraním mezi klasickými měřicími systémy a IoT (Internet Of Things - Internet věcí) aplikacemi.

Jak již bylo zmíněno v rámci kapitoly 1.1 výše, Datový koncentrátor by měl být přínosem primárně pro mobilní měření, kde právě konvenční měřicí systémy použít nelze.

Na druhé straně v rámci oblasti IoT je možné najít zařízení alespoň vzdáleně podobného typu. Většinou se však jedná o jedno zařízení, která nejsou nikterak modulární a jsou určeno pro jednoúčelová měření, přičemž princip vyvíjeného Datového koncentrátoru je právě založen na modularitě, kdy by měl umožnit průběh jakéhokoliv měření pomocí k tomu dostupných snímačů s definovaným rozhraním.

Dalším důvodem proč je vhodné vytvořit takovéto zařízení od úplného začátku je, že pokud se jedná o nějaké komerční zařízení většinou jde proprietární platformu s konkrétním určením, na kterou lze velmi těžko nebo vůbec navázat.

Příkladem za všechny může být zařízením ZM6380 Data Concentrator Unit, které od chytrých měřičů spotřeby elektrické energie koncentruje data a posílá je na privátní server. Jedná se však o velmi drahé průmyslové zařízení, které není určeno pro mobilní měření. Dále je toto zařízení primárně zaměřeno pouze na jeden daný typ měření. [2]

V případně otevřených platforem, projektů a zařízení s touto tématikou v rámci IoT nejsou tato řešení prakticky modulární, kdy nejčastějším řešením je vytvoření potřebných snímačů pro dané měření, které spoléhají pouze na cloudové prostředí a nemají vlastní lokální úložiště. Zde se k problému chybějící modularity taktéž přidává další omezení, kdy s takto postaveným systémem nejsou prakticky proveditelná měření s malou časovou periodou mezi jednotlivými měřicími body. Pokud již nějaká řešení modulární jsou, mají často velmi specifickou strukturu, na kterou je velmi těžké navázat a dále nad ní stavět.

Proto bylo rozhodnuto o vybudování takovéhoto modulárního zařízení, které má jednotné a jasně definované rozhraní, přesně danou strukturu, a bude se tak jednat o konsolidovaný a rozšiřitelný měřicí celek, který disponuje nadstavbou v podobě nadřazeného systému s možností analýzy dat již v průběhu měření.

1.4 Vydefinování hardwarových požadavků na Datový koncentrátor

V této sekci práce je pojednáno o konkrétním vydefinováním potřebných funkcionalit, které by mělo zařízení a jeho moduly obsahovat a splňovat. Následné požadavky tedy definují základní hranice pro výslednou realizaci konkrétního zařízení. Toto vydefinování pomůže v následujících kapitolách k výběru vhodného hardwarového a softwarového vybavení, ze kterých bude koncentrátor realizován.

Principy a koncepce jednotlivých modulů koncentrátoru již byly zmíněny v předešlé kapitole 1.2. Zde tedy dále nebude rozebírán jejich princip, avšak základní požadavky na tyto moduly a jejich funkcionality.

Datový koncentrátor by měl obsahovat vhodné a dostatečně výkoné jádro, které dokáže nejen obsluhovat všechny periferní moduly a zpracovávat a vhodně exportovat data, ale taktéž obstará podružné věci vzhledem ke koncentraci dat jako kontrolu stavu napájení a řešení neočekávaných hardwarových a softwarových situaci.

Lokální úložiště by mělo pojmout dostatečné množství dat ve výše zmíněných formátech (XML, JSON) a mělo by být dostatečně rychle přístupné, pro rychlý zápis a čtení naměřených dat ze senzorických jednotek. Taktéž by měl být umožněn export dat přímo z tohoto úložiště nezávisle na koncentrátoru, případně by měla existovat možnost takovéto úložiště z koncentrátoru vyjmout.

Modul reálného času by měl poskytovat přesný globální čas i přes případné uspání samotného zařízení. K dispozici by vždy mělo být aktuální datum a čas s přesností na celé sekundy. Ve spolupráci s funkčním výpočetním jádrem následně čas s přesností na jednotky milisekund.

Modul geografické polohy bude mít význam pouze pro venkovní měření avšak jak pro statická tak i pro dynamická. Od modulu geografické polohy se očekává přesnost určení aktuální zeměpisné polohy jak při statickém, tak i dynamickém pohybu lepší jak 5 m CEP (50 %) (Circural Error Probable - Pravděpodobnost kruhové chyby) (tedy CEP (50 %) $<$ 5 m) a určení přesnosti rychlosti pohybu za těchto podmínek s přesností méně než $\pm 1\, km/h$. [3]

Na synchronizační modul nejsou kladeny žádné specifické hardwarové, či softwarové požadavky. Měl by však zaručit distribuci aktuálního času směrem k senzorickým jednotkám, a to za účelem sjednocení času právě na senzorických jednotkách a datovém koncentrátoru.

Nízkoúrovňový komunikační modul by měl zaručovat jednoduchou a dostatečně rychlou komunikaci pro výměnu dat se senzorickými jednotkami. V tomto případě by se mělo jednat o typ bezdrátové komunikace jejíž komunikační frekvence by se měla nacházet v rámci pásem ISM (Industrial, Scientific And Medical Purposes - Pro účely vědy, techniky a zdravotnictví). Jedná se o rádiová pásma, která je možné na

daných frekvencích volně využít bez nutnosti vyřizování speciálního povolení a certifikace v případě dodržení výkonových pravidel vydaných Českým telekomunikačním úřadem.[4]

Vysokoúrovňový komunikační modul by měl zajistit komunikaci samotného koncentrátoru s nadřazeným systémem. V tomto případě by měl mít nadřazený systém podobu cloudového úložiště. Je tedy potřebné, aby komunikační modul dokázal zajistit připojení k internetu a vhodným protokolem komunikoval s cloudovým prostředím.

Pro zajištění přístupu k internetu takovéhoto mobilního zařízení se například nabízí vhodné řešení v podobě standardu GPRS a v rámci něj dále běžně používané komunikační protokoly ve světě IoT jako HTTP nebo MQTT. Výběr vhodného protokolu se samozřejmě odvíjí od komunikačního rozhraní nadřazeného systému, v tomto případě v budoucnu použitého cloudového prostředí.

1.5 Použité softwarové nástroje a prostředky

V této kapitole budou diskutovány softwarové nástroje a prostředky, které byly použity pro návrh, oživení a funkční zprovoznění Datového koncentrátoru. Budou zde blíže diskutovány důvody výběru níže vybraných prostředků a jejich přínos při samotné realizaci Datového koncentrátoru.

1.5.1 Eagle (Autodesk)

Eagle je software poskytovaný firmou Autodesk sloužící pro automatizaci elektronického návrhu - EDA, který umožňuje návrhářům desek plošných spojů (DPS) jednoduše vytvářet elektrická schémata a následně navrhnout rozložení a propojení jednotlivých elektrických komponent pro reálnou DPS. To vše za pomocí rozsáhlých knihoven obsahujících nejrůznější elektrické komponenty.[5]

Tento software je dostupný v omezené verzi pro každého jedince, limitace však spočívá v možnosti vytvořit pouze dvouvrstvou desku plošného spoje s plochou nepřesahující $80\,cm^2$. Samotné elektrické schéma taktéž může obsahovat nanejvýš dva listy obsahující samotné zapojení.[5]

Při návrhu samotné DPS Datového koncentrátoru byla využita studentská verze softwaru, která není svázána tak striktními pravidly, a bylo tak možné vytvořit přehledné elektrické schéma o šesti listech a DPS s výslednou plochou cca $100\,cm^2$.

Při samotném výběru návrhového prostředku pro vytváření DPS existuje v dnešní době přirozeně mnoho možností. Příkladem může být otevřený software KiCad EDA nebo dokonce volně dostupný online nástroj EasyEDA. Výběr softwaru Eagle se tedy zprvu může jevit nelogickým vzhledem k výše zmíněným neomezením, avšak

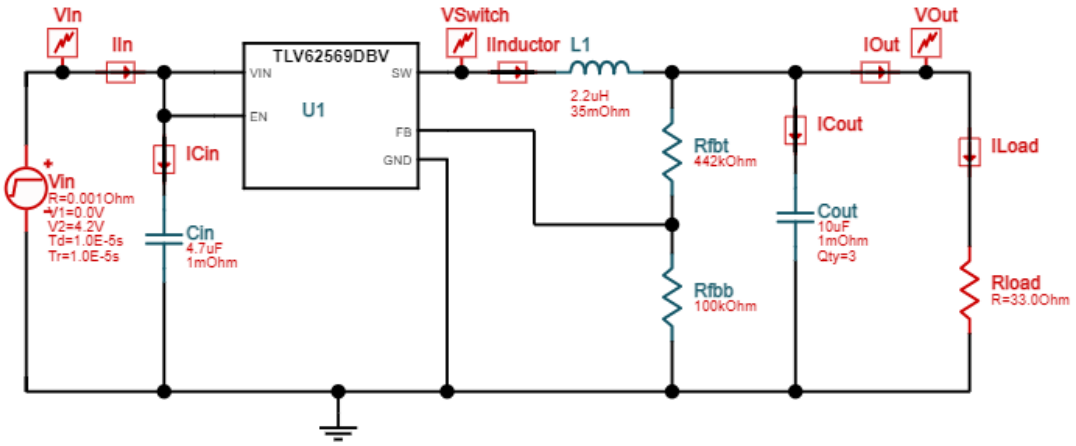
uvážíme-li skutečnost, že s vytvářením různých návrhu DPS v softwaru Eagle máme v minulosti již několik praktických zkušeností, dále pro námi vytvářené zařízení nás v návrhu omezení studentské licence nijak nelimitují a že s výběrem softwaru Eagle, jako prakticky nejrozšířenějšího EDA návrhového nástroje, získáme zázemí v podobě kvalitní AutoDesk podpory a široké uživatelské komunity, zdá se nakonec volba softwaru Eagle velmi rozumná a praktická.

1.5.2 Texas instruments - Webench® Power Designer

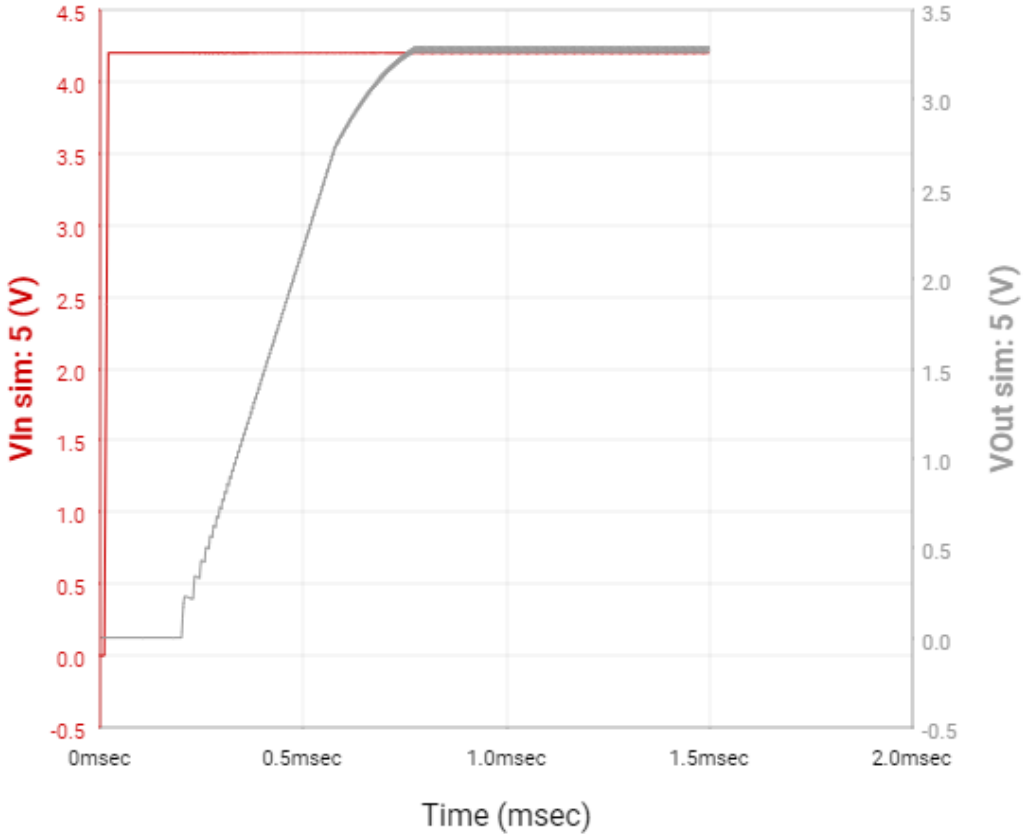
V návrhu zapojení desky plošných spojů pro datový koncentrátor byla pro DC-DC sestupný měnič, o kterém bude řeč v rámci kapitoly 2.1.7, využita komponenta od firmy Texas Instruments. Výhodou při použití komponent od Texas Instruments není pouze jejich kvalitní zpracování a dokumentace obsahující například doporučeného rozvržení přidružených komponent, nýbrž i fakt, že Texas Instruments poskytuje právě k výkonovým měničům i online návrhový nástroj "Webench® Power Designer" sloužící k výběru vhodného obvodu pro dané hardwarové řešení.

Tento nástroj umožňuje uživateli vybrat vhodnou komponentu na základě vstupně-výstupních požadavků, následně nabídne vhodnou obvodovou realizaci na základě zvolené komponenty a požadovaných vstupů a výstupů a taktéž umožní uživateli verifikovat chování dané obvodové realizace pomocí simulací. Uživatel tak může například získat představu o časových odezvách integrovaného obvodu či přenosnosti ustálených hodnot na základně parametrů připojené výstupní zátěže. Výstupy ze simulací lze samozřejmě exportovat v různých formátech (CSV, XLSX, JSON) pro následné další zpracování.[6]

V tomto případě byl tento nástroj použit pouze pro následnou verifikaci a simulaci již předtím namyšleného hardwarového řešení v podobě integrovaného obvodu TLV62569. Na obr. 1.3 lze tedy nahlédnout na verifikační obvodové schéma v nástroji Power Designer, na obr. 1.4 pak následně na výstupní graf samotné simulace.



Obr. 1.3: Verifikační obvodové schéma v nástroji Webench® Power Designer [6]



Obr. 1.4: Simulace měniče TLV65269 ve Webench® Power Designer [6]

1.5.3 PlatformIO IDE pro VS Code

Jedná se o jedno z hojně využívaných open-source IDE (Integrated Developmend Enviroment - Integrované vývojové prostředí) v rámci nástroje Microsoft Visual Studio Code pro vývoj embeded systémů. Jde se o uživatelsky přívětivé a rozšiřitelné integrované vývojové prostředí se sadou nástrojů pro profesionální vývoj aplikací, které poskytuje moderní a výkonné funkce pro zrychlení a zjednodušení vytváření systémů pro vestavěná zařízení.[7]

Doplněk PlatformIO do VS Code poskytuje editor zdrojového kódu pro nejrůznější platformy, dále inteligentní dokončení kódu na základě typů proměnných, definic funkcí a závislostí knihoven. Možnost práce na vícero projektech zároveň se snadnou navigací ve zdrojovém kódu daného projektu s vícero otevřenými navigačními panely a okny. PlatformIO taktéž podporuje jednoduchou a intuitivní práci a integraci softwarových knihoven.[7]

PlatformIO podporuje 41 vývojových platforem (jako například Atmel AVR, Espressif32), nabízí vývoj aplikací ve 23 různých frameworcích (Arduino, Espressif IoT Development Framework) a podporuje cirka 924 typů mikroprocesorových čipů, či vývojových desek.[7]

1.5.4 JLC-PCB

Pro realizaci Datového koncentrátoru je také nutno vzít v potaz jakým způsobem bude vyrobena samotná DPS realizující vodivé cesty Datového koncentrátoru. Vzhledem k náročnosti desky nebo případné neprofesionalitě zapřičiňující možnou nefunkčnost desky, nebyla výroba desky vlastními silami nikdy opravdovou možností. Proto je nutností nechat si takovouto DPS vyrobit na míru externě u nějakého výrobce.

Výběr vhodného výrobce pro samotnou výrobu desek plošných spojů není nijak složitou záležitostí, neboť na trhu je nepřeberné množství výrobců nabízejících vytvoření zakázkových DPS na základě lehce generovatelné dokumentace z EDA softwarů (Dokumentace DPS je předávána výrobci v podobě tzv. Gerber files). Hlavními kritérii na výrobu u takovéto desky je samozřejmě cena, výrobní kvalita a dosažitelnost v časovém horizontu, který je zákazník ochoten akceptovat.

V tomto případě byl výběr výrobce veden nejen výše zmíněnými kritérii na výrobu DPS, ale taktéž již předešlými osobními zkušenostmi s externí dodávkou DPS na míru a doporučením pro potřeby třetích osob. Na základě výše uvedených kritérií byl tedy vybrán výrobce JLCPCB, který je jeden z předních světových producentů prototypových desek plošných spojů.

Výrobce je schopen dodat až šestivrstvé plošné spoje různých velikostí od minimální 6 mm × 6 mm do maximální 400 mm × 500 mm. V případě potřeby je možné

u výroby DPS taktéž navolit si dle vlastních požadavků barvu, tloušťku, materiál kovových plošek pro uchycení SMD komponent (Surface Mount Device - Součástka určena pro povrchovou montáž), či si dokonce vyžádat teplotní test pro Automotive DPS.[8]

Tento výrobce byl taktéž vybrán z důvodu možnosti nechat si celou DPS nebo její určitou část profesionálně osadit vybranými SMD komponenty, což v konečném důsledku může zákazníkovi ušetřit čas a případně i finanční prostředky spojené s jeho vlastní výrobou a řešením problémů vzniklých v jejím průběhu.[8]

Posledním a taktéž důležitým parametrem je výrazná jednoduchost, intuitivnost a možnost kontroly DPS při jejím zadáváním do výroby v rámci prostředí JLC-PCB.

2 Hardware

Následující kapitola se zabývá výběrem vhodných hardwarových prostředků pro realizaci Datového koncentrátoru, elektrickým návrhem koncentrátoru a jeho realizací v podobě návrhu plošného spoje, osazením vybranými komponenty a oživením celé hardwarové realizace.

2.1 Výběr vhodných hardwarových modulů

Níže na obr. 2.1 lze vidět konkrétní návrh jednotlivých modulů realizující hardware koncentrátoru. Jak si lze všimnout, návrh vychází z koncepce Data koncentrátoru probírané v rámci kapitoly 1 a z principiálního blokového schématu vyobrazeného na obr. 1.2.

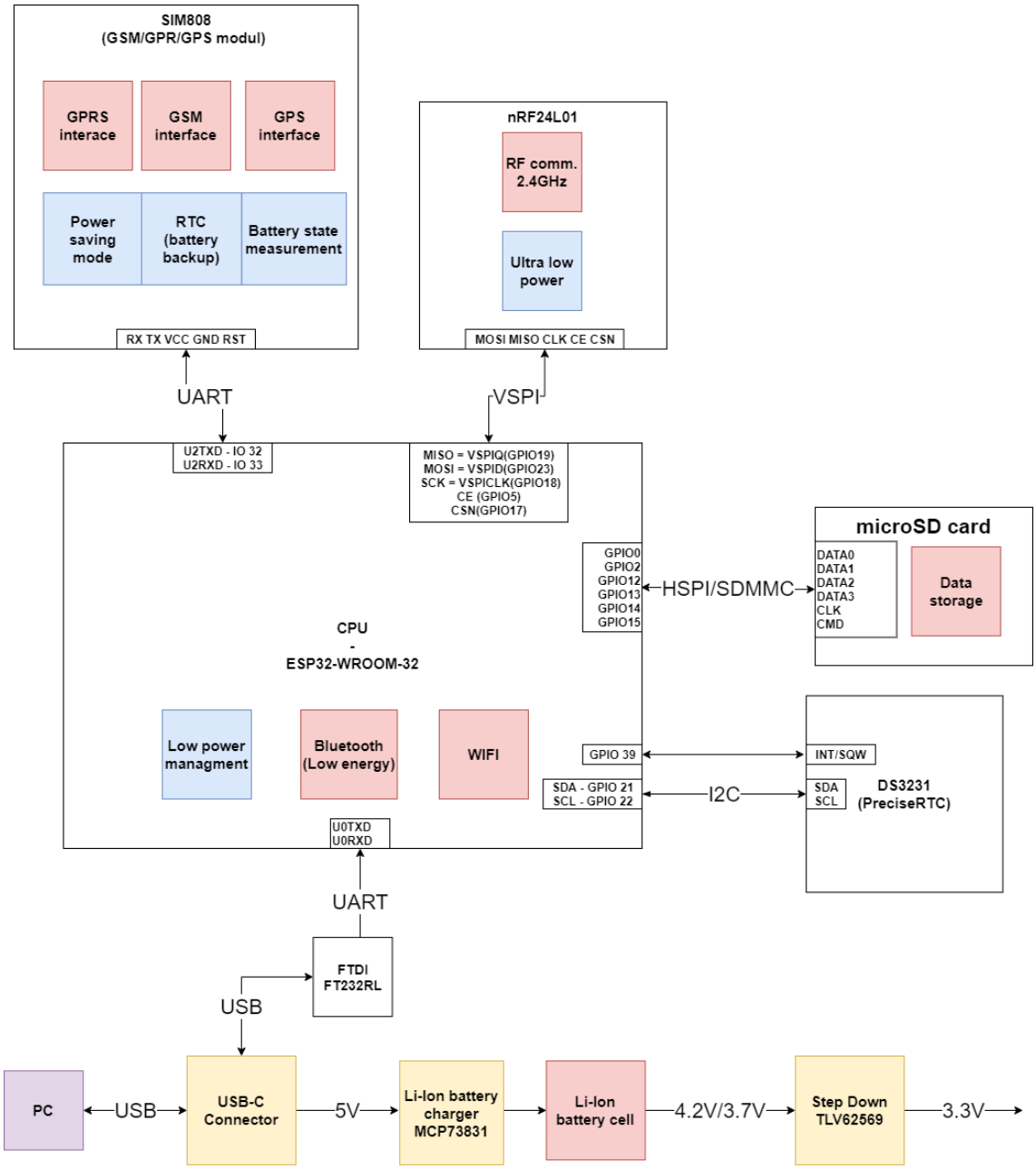
Všechny zde navržené moduly budou dále podrobněji rozebrány v následujících kapitolách, stejně jako důvody jejich výběru, o nichž bude vedena diskuse. Zde je uveden pouze jejich stručný popis a diskutovány jejich základní funkcionality a vlastnosti.

Zaměřme se však tedy na obr. 2.1. Schéma obsahuje jak jednotlivé navržené hardwarové prvky s popisem jejich nejpodstatnějších funkcionalit pro funkci koncentrátoru, tak i jednotlivá komunikační rozhraní spojující výpočetní jádro s těmito periferiemi. Právě na základě takových informací, jako jsou typy a počty použitých sběrnic ke zvoleným periferním modulům, které určují počet pinů mikroprocesoru potřebných pro komunikaci a ovládání právě těchto periferií, byl vybrán MCU (Microcontroller Unit - Mikrokontrolér) modul ESP32-WROOM-32. Ten tvoří výpočetní jádro celého systému a s jednotlivými moduly komunikuje a poveluje je.

Dále je na obrázku možné vidět rádiový komunikační modul nRF24L01. Jedná se o rádiový modul komunikující na frekvenci $2,4\,\mathrm{GHz}$ s relativně malou spotřebou a s komunikačním dosahem ve volném prostoru v řádech jednotek až desítek metrů, podle rychlosti komunikace. Totožný rádiový modul bude umístěn taktéž na jednotlivých senzorických jednotkách. Zde zvolený rádiový modul komunikuje s MCU modulem pomocí sběrnice SPI.

Z principu nejdůležitějším periferním modulem je v tomto případě datové úložiště, zde realizované v podobě microSD karty. Tento typ úložiště umožňuje rychlou komunikaci pomocí SPI nebo přes nativní rozhraní SD/SDIO/MMC a dostatečně velkou kapacitu úložiště v řádu jednotek až desítek Gigabajtů.

Pro udržení aktuálního času byly zvoleny hodiny reálného času v podobě čipu Maxis DS3231SN, jež obsahují vlastní vnitřní oscilátor a vykazují nejlepší časovou stabilitu v rámci svojí kategorie (ve stejné cenové relaci). Tyto hodiny pak s mikrokontrolérem komunikují pomocí sběrnice I2C.



Obr. 2.1: Konkrétní blokový návrh Datového koncentrátoru

Modul SIM808 v sobě kombinuje dvojí funkci, a to funkci GPS modulu a GPRS modulu. Má tedy za úkol jak zprostředkovávat zisk aktuální geografické polohy ve formě GPS souřadnic, tak zprostředkovat odesílání dat pomocí GPRS protokolu a sítě do cloudového prostředí připojeného taktéž do internetu. Samotný modul disponuje i dalšími přidruženými vlastnostmi, které budou zmíněny posléze. K výpočetnímu jádru je modul připojen pomocí sběrnice UART.

Vyobrazený napájecí modul zařízení se skládá z lineárního integrovaného ob-

vodu MCP73831 pro nabíjení jednočlánkových Li-Ion akumulátorů, jednočlánkového lithium-iontového akumulátoru typu Samsung SDI INR18650-35E a ze spínaného regulátoru snižující napětí TLV62569. Obvod MCP73831 realizuje nabíjecí režim konstantní proud/konstantní napětí potřebný pro nabíjení lithium-iontových akumulátorů. Jednočlánkový akumulátor následně poskytuje samostatně napájení modulu SIM808, který požaduje napájecí napětí $3,4\,\mathrm{V} - 4,4\,\mathrm{V}$ a následně napájení pro všechny ostatní moduly včetně mikroprocesoru pomocí vysoce efektivního spínaného regulátoru TLV62569. Ten snižuje napětí jednočlánku, které je typicky v rozmezí $3,4\,\mathrm{V} - 4,4\,\mathrm{V}$ na stabilní požadovanou hodnotu $3,3\,\mathrm{V}$.

Pro možnost programování samotného mikroprocesoru a následné možnosti komunikace mezi počítačem a mikroprocesorem byly zvoleny sběrnice USB, UART a čip FT232RL, který zprostředkovává rozhraní mezi těmito dvěma sběrnicemi. Dále byl pro připojení počítače pomocí USB sběrnice použit standardní průmyslový konektor USB typu C.

Schéma na Obr. 2.1 však neobsahuje všechny parametry a podrobnosti pro funkci koncentrátoru. Tyto podrobné aspekty budou diskutovány v rámci elektrického schématu Datového koncentrátoru v kapitole 2.3.

2.1.1 Modul nRF24L01 - Nízkoúrovňové komunikační rozhraní

Pro komunikaci mezi jednotlivými snímači a datovým koncentrátorem byl zvolen modul nR24L01 SMD osazený čipem nRF24L01+ od Nordic Semiconductors (Obr. 2.2). Modul poskytuje rádiovou komunikaci mezi více zařízeními s nastavitelnou frekvencí od $2,4\,\mathrm{GHz}$ do $2,525\,\mathrm{GHz}$.[10]

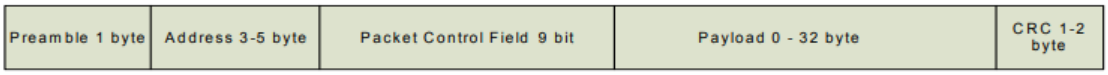
Obr. 2.2: Fotografie modulu nRF24L01 mini [9]

Jak již bylo zmíněno, jedná se o frekvenci, která je obsažena v rámci ISM pásem. Tuto rádiovou frekvenci je možné volně využívat stejně jako například frekvence

433 MHz a 868 MHz. Při výběru bylo taktéž uváženo použití těchto frekvencí, které disponují v případě frekvence 868 MHz řádově větším dosahem až 800 m, avšak taktéž nižší komunikační rychlostí do 100 kb/s. Podíváme-li se na možnosti komunikačních rychlosti modulu nRF24L01+ jsou v katalogovém listu popsány režimy 250 kb/s, 1 Mb/s, 2 Mb/s, což značí výrazně rychlejší komunikační rychlosti. Předpokládaný dosah modulu je při maximálním výkonu antény a nejvyšší komunikační rychlosti v řádech jednotek metrů, při nejnižší komunikační rychlosti však až 80 m za ideálních podmínek. Nutno podotknout, že moduly pro frekvence 868 MHz a 433 MHz byly řádově dražší než nRF24L01+ nebo podporovaly pouze point-to-point, či simplexní režim komunikace.[10]

Dalším důvodem pro výběr modulu nRF24L01 je, že nativně obsahuje linkovou vrstvu "Enhanced ShockBurst™"(ESB), která poskytuje automatický "packet handling", tedy skládání komunikačního paketu (viz obr. 2.3), včetně synchronizační preambule, adresy příjemce, informace o vnitřně přenášených datech, samotná přenášená data a kontrolní součet, který může být 8, či 16 bitový. Dále tato linková vrstva podporuje automatickou režiji okolo paketu, tedy potvrzení přijatého paketu, či jeho znovu odeslání. V neposlední řadě podporuje taktéž automatickou extrakci přenášených dat z přijatého paketu.[10]

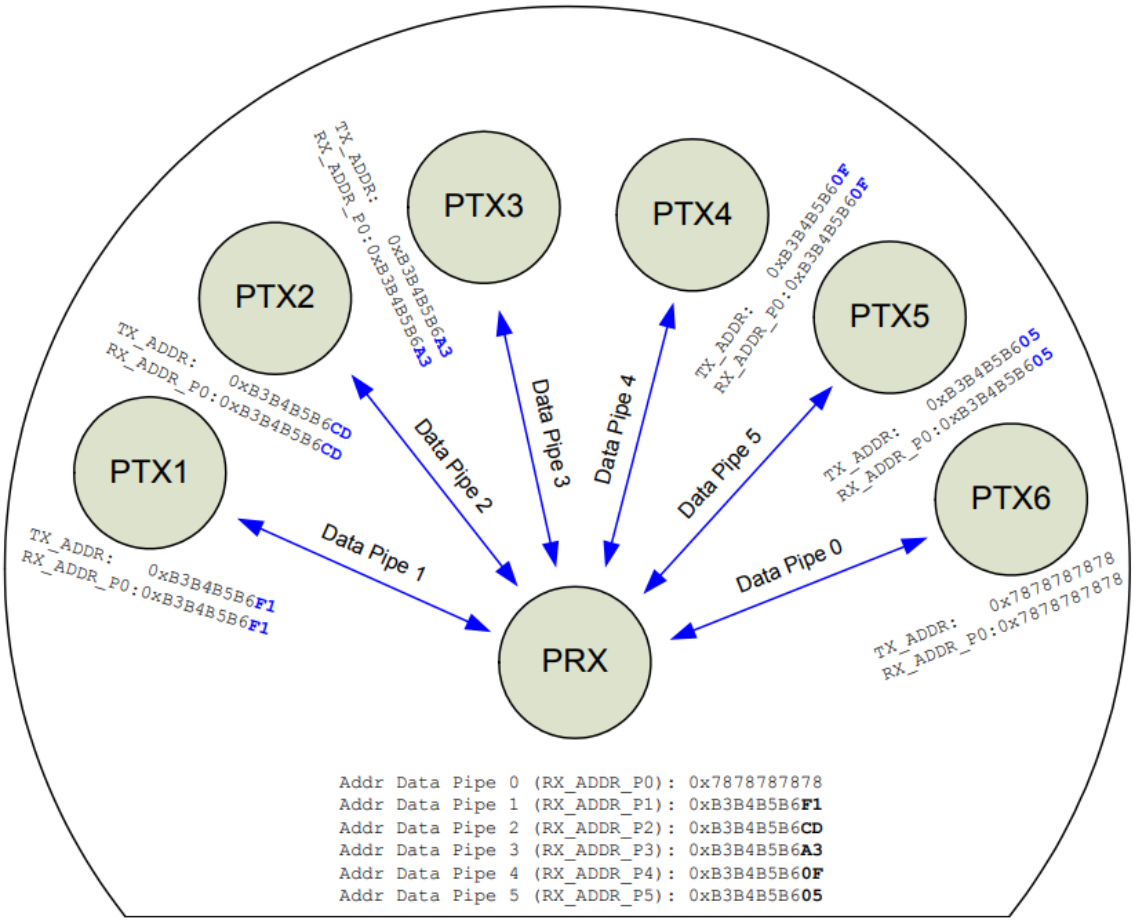
Mimo to, že "Enhanced ShockBurst™"taktéž nabízí možnost komunikovat pakety s konstantní velikostí přenášených dat, je možné komunikovat v režimu s proměnlivou velikostí přenášených dat (0-32 Bytů).[10]



Obr. 2.3: Formát paketu na linkové vrstvě modulu nRF24L01 [10]

"Enhanced ShockBurst™"podporuje "MultiCeiveir™", což je spojení jednoho zařízení (primárního přijímače) až se šesti dalšími zařízeními (primárními vysílači) pomocí šesti komunikačních kanálů, neboli "pipes". Jedná se tedy o topologii typu hvězda, kdy primární přijímač může buď poslouchat na všech kanálech zároveň nebo je jednoho z nich data zapisovat.[10]

Z obrázku 2.4 si lze taktéž všimnout, že pro komunikaci s jedním zařízeními v rámci ESB je, třeba sada dvou adres vysílací a přijímací. Tyto adresy mohou být totožné, ovšem mohou se i lišit, kdy primární přijímač má sadu přijímacích adres lišící se v posledních osmi bitech a dále každý primární vysílač má vlastní unikátní adresu. Takto vzniklé sady dvojic adres umožňují i případnou komunikaci mezi primárními vysílači a vedou k úplnému popisu komunikačních zařízení. Takovéto sady



Obr. 2.4: Komunikační topologie modulu nRF24L01 v rámci MultiCeiveir™ [10]

adresových dvojic byly využity v rámci implementace komunikace se senzorickými jednotkami v kapitole 3.3.4

Dále je nutné poznamenat, že běžně jsou využívány pouze komunikační kanály jedna až pět, neboť paměťovým přijímací adresový registr pro kanál 0 sdílí i funkci s paměťovým vysílacím adresovým registrem. [10]

V případě potřeby více komunikujících zařízení (snímačů), lze softwarově vytvořit síť typu strom, kdy na nejvyšší pozici je hlavní uzel a k němu může být připojeno až 5 potomků. každý potomek může mít dále dalších 5 potomků, přičemž celková síť může mít teoreticky až 3125 uzlů. K tomu je však potřeba součinnost jednotlivých potomků a nadstavbové programové vybavení.

2.1.2 Modul SIM808 - Vysokoúrovňové komunikační rozhraní

Pro komunikaci MCU modulu s nadřazeným systémem byl vybrán modul SIM808 (viz obr. 2.5). Hlavním důvodem pro výběru právě tohoto modulu bylo to, že sa-

motný koncentrátor má mít možnost získat svojí aktuální geografickou polohu v podobě GNSS souřadnic a taktéž mít možnost komunikovat uložené data do nadřazeného systému, v tomto případě cloudového prostředí, tedy je třeba zajistit připojení koncentrátoru k internetu.

Původní myšlenkou bylo použití separátních modulů, tedy GNSS/GPS modul v podobě NEO-6, NEO-7 a pro připojení k síti GSM/GPRS modul SIM800L. Při takovéto konstalaci bychom však potřebovali dvě komunikační sběrnice, každou pro jeden modul. Dále modul SIM800L je konstrukčně navržen tak, že má několik vodivých pinů ze spodní strany pouzdra, takže modul by nebylo možné ručně zapájet. Řešením by mohlo být použití výše zmíněných modulů ve formě vývojových kitů, avšak to se vzhledem k jejich velikosti jeví jako velmi nepraktické řešení.

Byl tedy vybrán modul SIM808, který právě slučuje obě funkcionality, tedy chová se jak GSM/GPRS modul tak i GPS modul.

Obr. 2.5: Horní a dolní pohled na modul SIM808 [11]

Z pohledu GSM/GPRS modul SIM808 poskytuje komunikaci v rámci mobilní sítě na frekvencích 850/900/1800/1900MHz. Maximální rychlost komunikace pomocí protokolu GPRS je 85,6 kb/s pro upload i download dat. V rámci modulu je taktéž nativně podporován protokol TCP/IP pro možnost připojení zařízení k internetu a protokol HTTP pro aplikační vrstvu.[11]

V rámci GPS modul SIM808 nabízí přesnost horizontální polohy menší než 2,5 m CEP (50 %). Přesnost určení vertikální polohy v rámci katalogového listu není uvedena, avšak bylo empiricky zjištěno, že chyba určení vertikální polohy koreluje s chybou určení horizontální polohy a lze ji odhadnout jako 1,7 násobek chyby určení v horizontální poloze. V tomto případě by tedy přesnost určení vertikální polohy, tedy přesnost určení nadmořské výšky v daném měřeném bodě, měla být lepší než 4,25 m.[11, 12]

Dále SIM808 v rámci funkce GPS slibuje přesnost určení rychlosti pohybu s přesností pod $0,1\,\mathrm{ms}^{-1}$ $(0,36\,\mathrm{km/h})$. Pro určení polohy modul využívá až 66 akvizičních

kanálů, které kontrolují příchod nových signálů z družic a až 22 kanálu v kontinuálním režimu sledující potvrzené signály. Obnovovací frekvence aktuální polohy je 5 Hz.[11]

SIM808 dále nabízí i další zajímavé periferie jako PWM výstup, vstupně-výstupní piny nebo audio výstup. Ty však v rámci Datového koncentrátoru potřebné nebyly a tudíž nebyly ani využity.[11]

Samotný modul komunikuje s mikroprocesorem přes sériové rozhraní UART.

2.1.3 MicroSD karta - Datové úložiště koncentrátoru

Jako lokální datové úložiště koncentrátoru byl zvolen způsobo ukládání dat na microSD kartu. Výběr tohoto typu úložiště byl zvolen na základě následujících vlastností.

Jedná se o typ nevolatilní paměti s dostatečnou úložnou kapacitou (viz níže v textu) a taktéž o typ úložiště, které je vyjímatelného typu a tedy umožňuje zpracování uložených dat i v rámci externího systému.

ESP32-WROOM-32 disponuje možností připojit microSD kartu pomocí pomocí dvou rozhraní, a to SPI nebo nativního rozhraní pro SD/SDIO/MMC karty. Pokud uvažujeme námi zvolenou microSD kartu je pro nás k dispozici v rámci nativního rozhraní 1-bitová a 4-bitová datová sběrnice a v rámci SPI pouze komunikační piny MISO a MOSI. Oba typy sběrnic mají základní frekvenci 20 MHz, pro vysokorychlostní karty 40 MHz. [13, 14]

ESP32 pracuje s SD kartami ve formátech FAT16 nebo FAT32. Omezení velikosti našeho úložiště tedy plynou z typu vybraného souborového uspořádání našeho úložiště.

V našem případě bude využito formátu FAT32. Omezení přichází vzhledem k našemu operačnímu systému (Windows), který dokáže přečíst data ve formátu FAT32 s maximální velikostí úložiště 32 GB. Dále formát úložiště FAT32 omezuje maximální velikost jednoho souboru na 4 GB. To však v našem případě není reálným omezením, neboť v rámci koncentrátoru budeme ukládat malé textové soubory v jednotkách až desítkách kilobajtů.[15]

Na základě výše zmíněného byla pro Datový koncentrátor vybrána SD karta typu SDHC ve formátu FAT32. Dále by tato karta měla být pro plné využití komunikační rychlosti sběrnice SPI nebo 1-bitové nativní sběrnice alespoň v rychlostní třídě 6 (6 MB/s). Dále doporučená a v zařízení použitá SD karta je v rychlostní třídě 10 (10 MB/s), nebo taktéž v třídě UHS 1, což je v tomto případě ekvivalent. Rychlost zápisu však v reálu bude jistě nižší neboť jsme limitování jak rychlostí samotné sběrnice, tak i implementací knihovny pro správu souborového systému (viz obr 3.6)

a samotnou velikostí ukládaných dat, které v konečném důsledku ovlivňují měřenou délku zápisu.[16]

2.1.4 DS3231SN - Hodiny reálného času koncentrátoru

Pro udržení aktuálního času Datového koncentrátoru byl zvolen čip DS3231SN jako velmi přesný RTC (Real Time Clock - Hodiny reálného času) integrovaný obvod obsahující vlastní teplotně kompenzovaný oscilátor. Chyba udržovaného času je výrobcem udávaná $\pm 2\,\text{ppm}$ v rámci teplotního intervalu 0 °C - 40 °C, dále pak $\pm 3,5\,\text{ppm}$ v rámci intervalu -40 °C - 85 °C. [17]

Uvažujme tedy nejhorší možný případ, tedy $\pm 3,5\,\text{ppm}$ v rámci jedné a dvaceti čtyř hodin:

$$\Delta_{\text{t=1 h}} = \delta_{\text{t}} \times t = \frac{3,5}{10^6} \times 3600 = 0,0126\,\text{s} = 12,6\,\text{ms} \tag{2.1}$$

$$\Delta_{\text{t=24 h}} = \delta_{\text{t}} \times t = \frac{3,5}{10^6} \times 86400 = 0,3024\,\text{s} = 302,4\,\text{ms} \tag{2.2}$$

Kde Δ_{t} je absolutní chyba za daný časový interval, δ_{t} je relativní chyba udaná výrobcem a t je uvažovaný čas od nahrání námi považované správné hodnoty do obvodu DS3231SN.

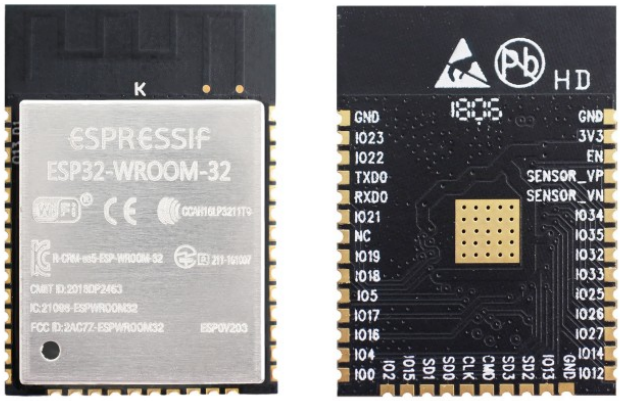
V rámci výpočtu lze vidět, že v rámci intervalu jedné hodiny jsme schopni udržet původně správný čas s přesností na desítky milisekund za celý den v rámci stovek. Důležité je podotknout, že absolutní chybu měření lze v tomto případě libovolně zmenšovat pomocí zmenšení periody pravidelné korekce správného času. Takovouto korekci v rámci koncentrátoru bude možné vykonat, neboť informaci o nejpřesnějším aktuálním čase lze extrahovat pomocí připojení přes GPRS, či z GPS rozhraní v rámci výše zmíněného modulu SIM808.

DS3231SN s datovým koncentrátorem komunikuje pomocí sběrnice I2C. Ke sběrnici je z integrovaného obvodu taktéž paralelně vyvedeno jednopinové spojení pro generování přesných pravidelných hodinových pulzů o frekvencích 1 Hz, 1024 Hz, 4096 Hz, či 8192 Hz. Toto spojení lze taktéž použít pro vyvolání přerušení na základě dvou uživatelsky nastavitelných alarmů.[17]

2.1.5 ESP32-WROOM-32 - Výpočetního jádro koncentrátoru

ESP32-WROOM-32 je víceúčelový výkoný Wi-Fi+BT+BLE MCU modul pro obecné použití (viz obr. 2.6). Jádrem tohoto modulu je mikrokontrolér ESP32-D0WDQ6. Tento mikrokontrolér je osazen dvoujádrovým 32-bitovým mikroprocesorem Xtensa® LX6 s nastavitelným taktováním od 80 MHz do 240 MHz. Taktéž je zde přítomen nízko spotřebový co-procesor sloužící k provádění výpočetně nenáročných aplikací

jako například monitorování periferií. Mikrokontrolér taktéž mimo paměti ROM a SRAM určené pro mikroprocesor disponuje zabudovanou SPI flash pamětí o velikosti 4 MB. [18, 19]

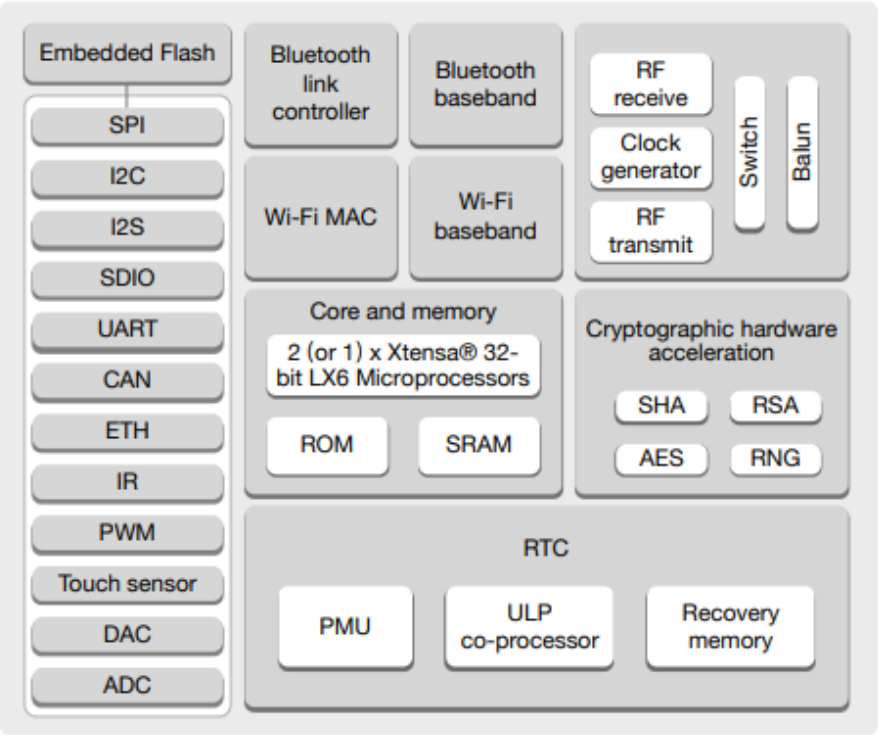


Obr. 2.6: Foto modulu ESP32-WROOM-32 [22]

Na obrázku 2.7 níže, který obsahuje funkční blokové schéma mikrokontroléru ESP32-D0WDQ6 lze dále vidět, že samotný mikrokontrolér nabízí velkou řadu diverzitních periferií, které jsou v rámci Datového koncentrátoru hojně využívány a na základě kterých byl právě tento MCU modul taktéž vybrán jako vhodné výpočetní jádro systému.

Celý MCU modul ESP32-WROOM-32 je tedy zkompletován z DPS, na které je osazen mikrokontrolér ESP32-D0WDQ6, ten je dále doplněn o základní pasivní komponenty a ESD (Electrostatic Discharge - Elektrostatický výboj) ochranu. Dále je v rámci modulu obsažena zabudovaná anténa pro Wi-fi a Bluetooth rozhraní. Mimo anténu je modul doplněn o robustní plechové zapouzdření. Vyvedení pinů mikrokontroléru na okraje DPS a celková fyzická podoba modulu umožňuje jednoduché osazení tohoto modulu v rámci cílové zákaznické desky.

V rámci výběru mikrokontroléru, či MCU modulu pro výpočetní jádro Datového koncentrátoru byly taktéž uváženy jiné platformy od různých výrobců. Přímým konkurentem při výběru byly mikrokontroléry z rodiny STM32. V rámci ní je vedeno široké portfólio různě výkonných modelů. Nebyl však nalezen žádný, který by nativně nepodporoval Wi-Fi a Bluetooth. Další výhodou je právě již zmiňované robustní zakrytování modulu. V neposlední řadě výhodou při výběru modulu ESP32-WROOM-32 může být jeho široké rozšíření a použití, tedy výhoda široké základny uživatelů, která na dané platformě pracuje.



Obr. 2.7: Funkční blokové schéma mikrokontroléru ESP32-D0WDQ6 [19]

2.1.6 FT232RL - Komunikační rozhraní USB/UART

Jako základní programovací a komunikační rozhraní MCU modulu s PC je využíváno UART sběrnice (číslo 0) na straně modulu a USB sběrnice na straně PC. Mezi těmito sběrnicemi je nutno použít převodník, který bude komunikovaná data vhodně konvertovat vzhledem k typu dané sběrnice a umožní tak oběma zařízení i přes rozdílnost sběrnice komunikovat.

V tomto případě byl vybrán integrovaný obvod FT232RL od firmy FTDI. Jedná se o standardně používaný USB-UART převodník. Při doplnění FT232RL vhodnými převodníky úrovní může taktéž fungovat jako převodník USB-RS232, USB-RS485, či USB-RS422.[23]

Na straně USB je v rámci FT232RL podporován standard USB 2.0 Full Speed. Na straně UARTu je zajímavá možnost komunikovat na různých napěťových úrovních a to 1,8 V/ 2,8 V/ 3,3 V/ 5 V.[23]

Čip je typu "plug and play" a celá konverze USB-UART se odehrává na čipu. Ten je tedy při své instalaci na desku je již plně funkční a není tedy třeba žádné prvotní instalace potřebného firmwaru.[23]

Alternativou pro vytvoření rozhraní USB-UART na zákaznické desce mohou být integrované obvody CP2101 od Silcion Labs nebo PL2303 od firmy Prolific.

2.1.7 Napájení Datového koncentrátoru

Vzhledem k vybraným modulům a komponentám pro Datový koncentrátor je taktéž vhodné pozastavit se nad volbou napájení samotného koncentrátoru. Jedná se o mobilní zařízení, je tedy nutné využít baterie, či akumulátoru

Taktéž je nutné zohlednit a zhodnotit požadavky trvalých a špičkových odběrů jednotlivých modulů a příslušná požadovaná napětí.

Následující podkapitoly tedy zhodnocují výběr jednotlivých komponent pro napájení Datového koncentrátoru.

Akumulátor Samsung SDI INR18650-35E

Pro napájení Datového koncentrátoru byl zvoleno řešení v podobě jednočlánkového lithium-iontového akumulátoru, konkrétně akumulátor INR18650-35E s celkovou kapacitou 3400 mAh a nominálním napětím 3,6 V (maximálním 4,2 V, minimálním 2,65 V). [24]

Obr. 2.8: Fotografie dvou jednočlánkových akumulátorů INR18650-35 [25]

Prvním důležitým aspektem, který byl zmíněn výše, je proudová zatížitelnost takovéhoto zdroje, přičemž vybraný akumulátor INR18650-35E dokáže poskytnout kontinuální proud v hodnotě 8 A, pro špičkové odběry až 13 A.[24]

Akumulátor tedy plně vykryje proudové špičky modulu SIM808, který je vzhledem k požadovanému napětí připojen přímo k baterii a jehož špičkový proudový odběr může dosahovat až 2 A. Dále níže diskutovaný spínaný snižující měnič TLV62569 na 3,3 V (viz podkapitola 2.1.7), byl taktéž s rezervou dimenzován na maximální proudové odběry. Přičemž dle katalogových listů ESP32-WROOM-32 má minimální proudový odběr 500 mA, kumulativní výstupní proud až 1100 mA, modul nRF24L01 maximální spotřebu v nejnáročnějším provozním módu 13,5 mA, DS3231SN odběr maximálně 200 μA a čip FT232RL je napájen z přivedené sběrnice USB.

Na základě zmíněných hodnot lze jejich jednoduchým sečtením zjistit, že spínaný snižující měnič poskytující maximální proudový odběr až 2 A v kombinaci s akumulátorem INR18650-35E s dostatečnou rezervou tuto špičkovou proudovou spotřebu všech modulů pokryje.

Katalogový list akumulátoru sice pojednává o celkové kapacitě 3400 mAh, avšak je nutné uvážit, že při vybíjení akumulátoru dochází k postupnému poklesu napětí na jeho svorkách (viz obr. 2.9) a taktéž že pro napájení zařízení je nutné se pohybovat v určitých napěťových intervalech. Vzhledem k modulu SIM808 to znamená možný napájecí napěťový rozsah 3,4 V až 4,4 V a dále vzhledem k napájení ostatních modulů je nutné napětí 3,3 V vytvářeného pomocí TLV62569. Uvažme nejhorší případ, tedy odhadnutý maximální špičkový proudový odběr za měničem cirka 1,2 A a požadované výstupní napětí 3,3 V, dále tedy ohledně katalogového listu platí následující:

$$U_{\mathrm{IN(MIN)}} = U_{\mathrm{OUT}} + I_{\mathrm{OUT}} \times (R_{\mathrm{DS(OUT)}} + R_{\mathrm{L}}) \tag{2.3}$$

$$U_{\mathrm{IN(MIN)}} = 3,3\,\mathrm{V} + 1,2\,\mathrm{A} \times (100\,m\Omega + 35\,m\Omega) = 3,462\,\mathrm{V} \tag{2.4}$$

Kde $U_{\mathrm{IN(MIN)}}$ je minimální požadované napětí na vstupu měniče TLV62569, U_{OUT} je požadované výstupní napětí z měniče, I_{OUT} je námi odhadnutý maximální odebíraný proud z měniče, $R_{\mathrm{DS(OUT)}}$ je odpor sepnutého výstupního tranzistoru FET měniče a R_{L} je odpor námi použité cívky.

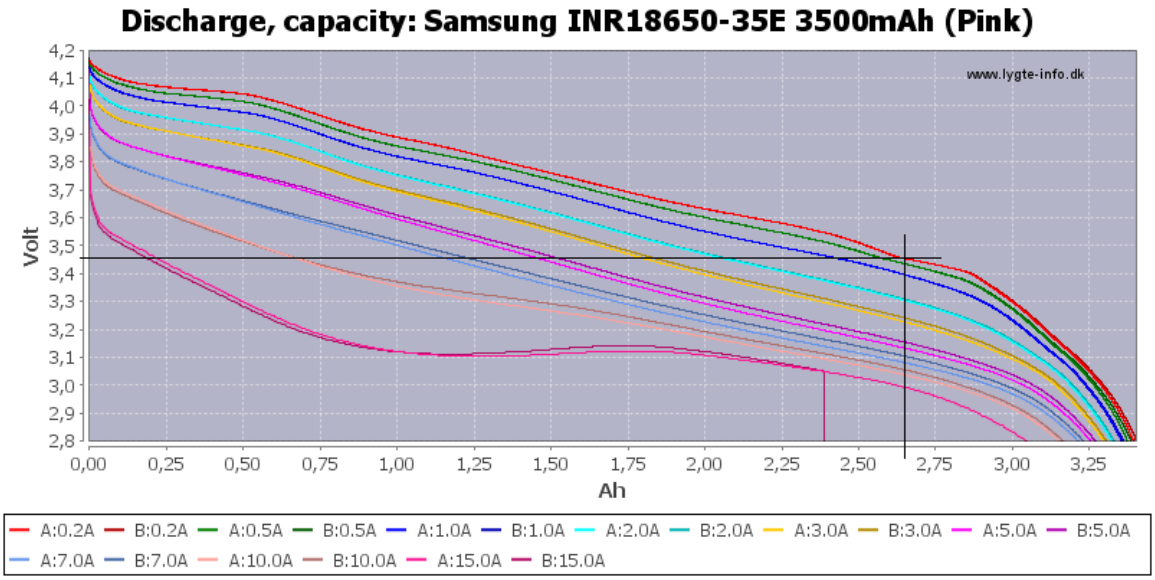
Z výpočtu a výše zmíněného tedy vyplývá, že je nutné operovat v intervalu napětí na akumulátoru v rozmezí 4,2 V až 3,462V. Pokud tyto hodnoty přeneseme do grafu na obrázku 2.9 a budeme uvažovat průměrný proudový odběr koncentrátoru v rámci 0,2 A vyplývá z grafu, že se v tomto případě dostáváme na využitelnou kapacitu nového akumulátoru o hodnotě přibližně 2,650 mAh.

Dále tedy opět předpokládejme průměrný proudový odběr 0,2 A, z něhož vyplývá následující výpočet doby provozu Datového koncentrátoru napájeného z plně nabitého akumulátoru INR18650-35E.

$$Q = I \times t \rightarrow T_{\mathrm{ON}} = \frac{Q_{\mathrm{Bat}}}{I_{\mathrm{Prum.}}} = \frac{2650\,\mathrm{mAh}}{200\,\mathrm{mA}} = 13,25\,\mathrm{h} = 13\,h : 15\,m : 00\,s \tag{2.5}$$

Kde Q je elektrický náboj, Q_{Bat} je uchovávaný náboj v akumulátoru nebo taktéž kapacita akumulátoru, I je proud, $I_{\mathrm{Prum.}}$ je odhadnutý průměrně odebíraný proud, t je čas a T_{ON} je doba po kterou je možné koncentrátoru napájet z akumulátoru.

Dále při použití jednočlánkového lithium-iontového akumulátoru je závěrem nutné zmínit, že výše uvedené výpočty jsou platné pro nový akumulátor, plně nabitý akumulátor v prostředí s okolní teplotou 23 ± 3 °C a dále relativní vlhkostí 65 %.[24]



Obr. 2.9: Graf vybíjení INR18650-35E v závislosti na proudovém zatížení [25]

Li-Ion akumulátory totiž obecně trpí teplotní závislostí, tedy pro jejich řádný provoz je nutné operovat v určitých teplotních oknech. V případně velmi nízkých teplot, typicky pod bodem mrazu, klesá těmto typům akumulátorů jejich kapacita a napětí, což je dáno růstem vnitřního odporu článku. Růst vnitřního odporu je způsoben nízkými teplotami, které způsobují snížení kinetiky reakcí a transportů iontů v rámci elektrod lithium-iontového článku.[26]

Stejně je tomu při velmi vysokých teplotách, kdy typicky již nad 50 °C dochází k urychlenému stárnutí článku a zkrácení jeho životnosti. Dále při dalším zvyšování teploty může docházet k rozpadu ochranné vrstvy mezi zápornou elektrodou článku a jeho elektrolytem, což opět vede ke ztrátě kapacity článku a ohřevu samotného akumulátoru a v krajních případech k jeho zničení.[26]

V tomto kontextu pojednává i katalogový list pro článek INR18650-35E, kdy jsou v něm definována provozní teplotní okna. Pro nabíjení článku jsou přípustné provozní teploty v rozmezí 0-40 °C, pro vybíjení pak -10-60 °C. Dále jsou zde uvedeny příklady využitelné kapacity při vybíjení při jednotlivých teplotách.[24]

Pro vybíjení byla pro článek INR18650-35 při teplotách 23 °C a 45 °C změřena jeho využitelná kapacita na 97 % z celkové kapacity, při teplotě 0 °C to bylo pouze 40 %. Ne jinak je tomu pro nabíjení, kdy opět při teplotách 23 °C a 45 °C bylo možné článek nabít na 100 % jeho původní kapacity, při 0 °C pouze na 60 %.[24]

TLV62569

Jak již bylo zmíněno výše v rámci napájení modulů, které vyžadují napětí 3,3 V, byl použit snižující měnič TLV62569 od firmy Texas Instruments s maximálním výstupním proudem 2 A. Jedná se o vysoce efektivní spínaný měnič s frekvencí spínaného výstupu 1,5 MHz. Z katalogového listu lze vyčíst, že pro hodnoty, které jsou v rámci napájení koncentrátoru při konverzi napětí uvažované, tedy U_{IN} v rozmezí 4,2 V - 3.462 V a $U_{\mathrm{OUT}} = 3{,}3$ V, se efektivita měniče nachází v rozmezí $92\,\% - 95\,\%$.[27]

Variabilní vstupní napětí z akumulátoru nečiní měniči žádný problém, přičemž vstupní napětí měniče U_{IN} může být v rozmezí 5,5 V až $U_{\mathrm{IN(MIN)}}$. Taktéž s tímto záměrem byl tento měnič vybrán.[27]

Korektnost návrhu měniče, na jehož zapojení lze nahlédnout v rámci elektrického schématu v příloze A, byla taktéž verifikována pomoci online nástroje "Texas Instruments Webench® Power Designer", který je podrobněji komentován v rámci kapitoly 1.5.2.

MCP73831

Jako poslední byl vybrán integrovaný obvod MCP73831 pro napájení Li-Ion a Li-Pol jednočlánkových akumulátorů. Jedná se o lineární nabíjecí obvod s konstantním proudem/konstantním napětím. Nabíjecí konstantní proud může být až 500 mA, konstantní napětí je v tomto případě 4,2 V. Vstupní napětí pro nabíjecí obvod 5 V je přivedeno ze sběrnice USB.[28]

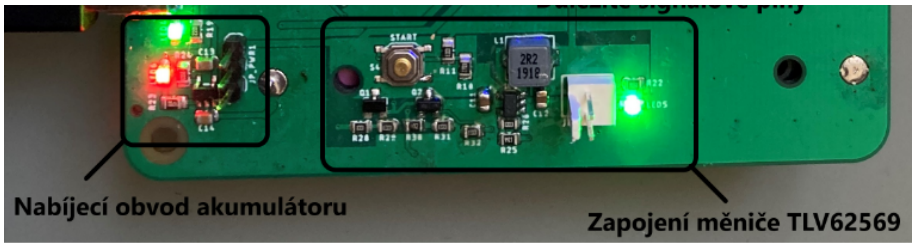
2.2 Realizace Datového koncentrátoru

V této kapitole bude stručně popsána již hotová hardwarová realizace Datového Koncentrátoru a její jednotlivé komponenty. Podrobnější rozbor navrženého elektrického schématu na základě kterého realizace vnikla a samotného návrhu desky plošných spojů budou rozebrány v následujících kapitolách 2.3 a 2.4.

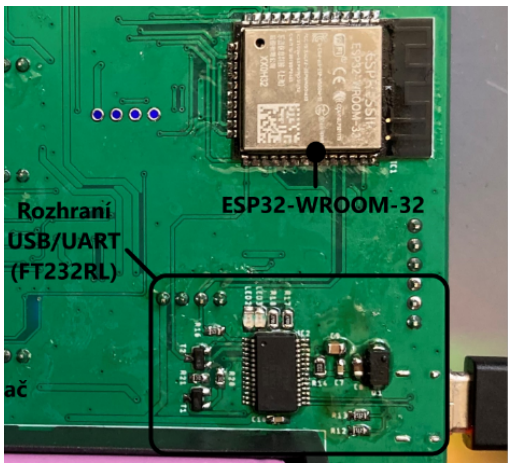
Jak již bylo zmíněno výše jedná se o zařízení, které je napájeno z Li-Ion akumulátoru. Na obrázku 2.10 lze vidět napájecí obvod a dále realizace sestupného měniče pro vytvoření napájecího napětí 3,3 V včetně tlačítka pro start zařízení. Napájení pro nabíjení akumulátoru je přivedeno z konektoru USB-C (viz obr. 2.14).

Dále na obrázku 2.11 lze vidět umístění výpočetního jádra v rámci desky včetně pro něj určeného rozhraní USB/UART pro komunikaci s PC. Dále je zde viditelné vyvedení konektoru USB-C pro připojení zařízení k PC.

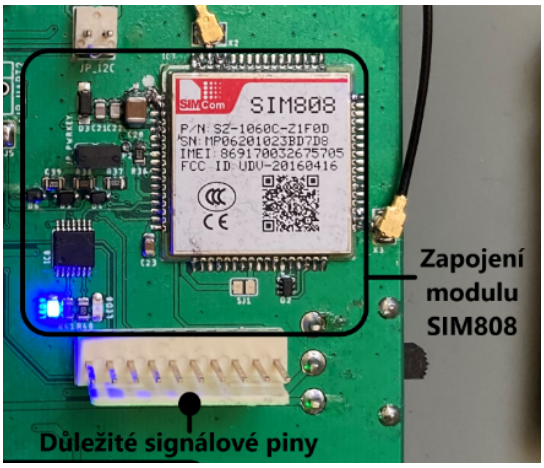
Na obrázku 2.12 je vidět připájený modul SIM808 s rozmístěním jeho obvodové realizace a s konektory pro připojení dvou pasivních antén (GPS a GSM/GPRS).



Obr. 2.10: Realizace nabíjecího obvodu a sestupného měniče

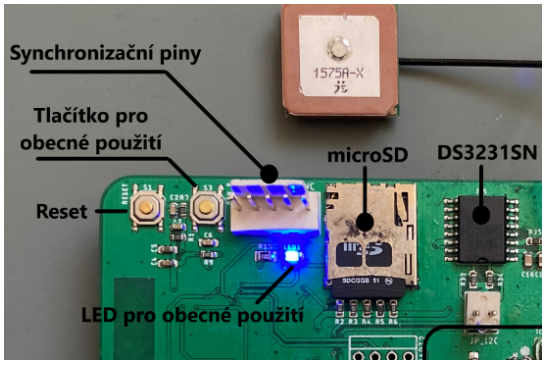


Obr. 2.11: Osazené výpočetního jádro spolu s čipem FT232RL

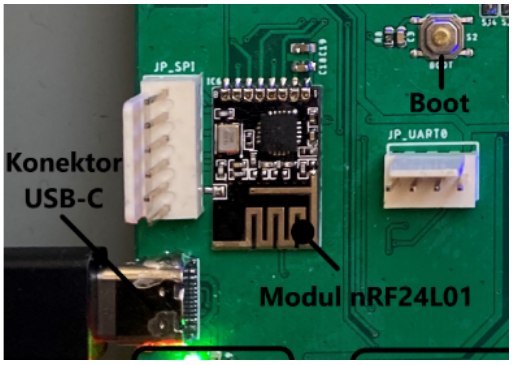


Obr. 2.12: Realizace modulu SIM808 a vyvedení signálových pinů

V obrázku 2.14 následně umístění komunikačního modulu nRF24L01 v rámci desky a obrázek 2.13 vyobrazuje umístění lokálního úložiště, obvodu DS3231SN a uživatelského rozhraní.

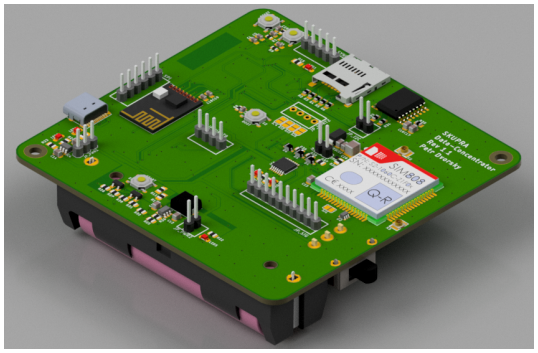


Obr. 2.13: Uživatelské rozhraní, SD slot a obvod DS3231

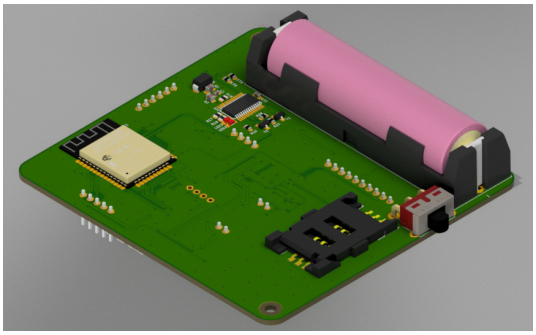


Obr. 2.14: Osazení modulu nRF24L01 a konektoru USB-C

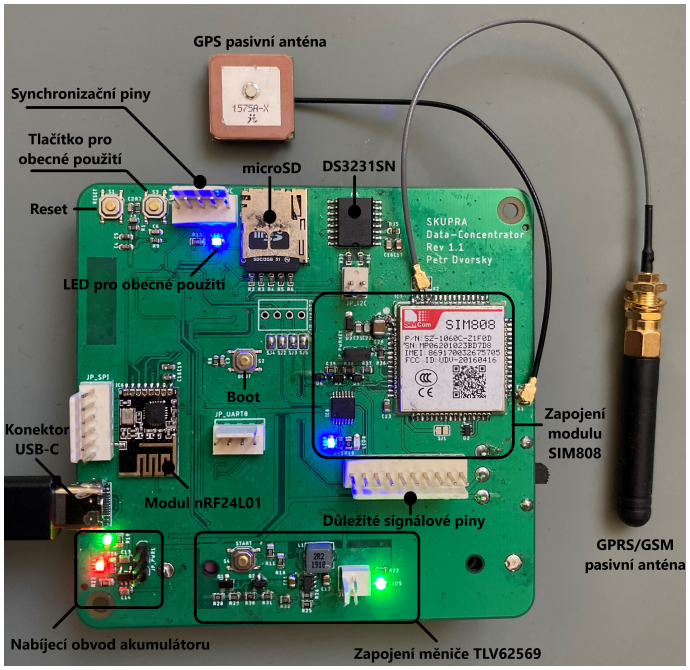
Níže na obrázcích 2.17 a 2.18, které byly podkladem pro předchozí, lze vidět kompletní reálnou a oživenou podobu desky Datového koncentrátoru a dále na obrázcích 2.16 a 2.15 3D model tohoto zařízení. Přiblížení jednotlivých obrázků je následně možno nalézt v příloze C a D.



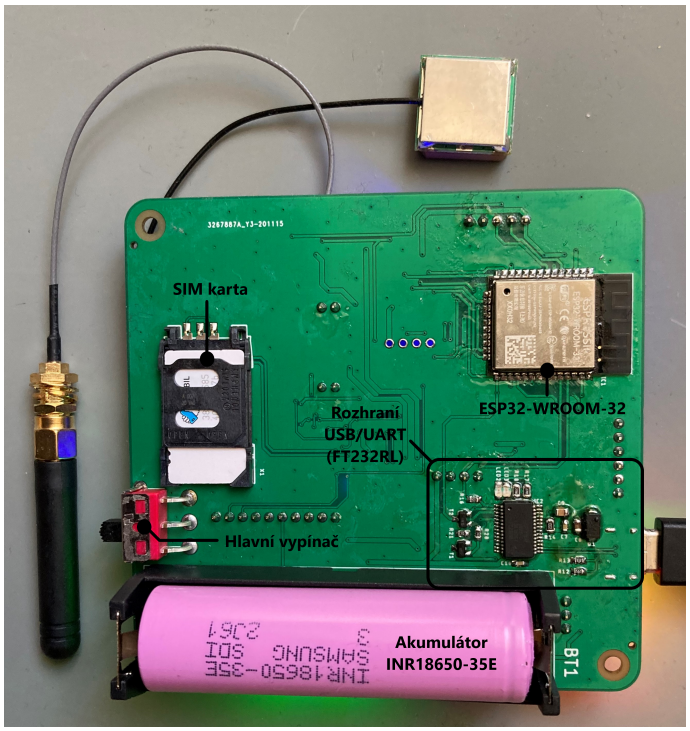
Obr. 2.15: 3D model DPS Datového koncentrátoru - Přední strana



Obr. 2.16: 3D model DPS Datového koncentrátoru - zadní strana



Obr. 2.17: Osazená DPS Datového koncentrátoru - Přední strana



Obr. 2.18: Osazená DPS Datového koncentrátoru - Zadní strana

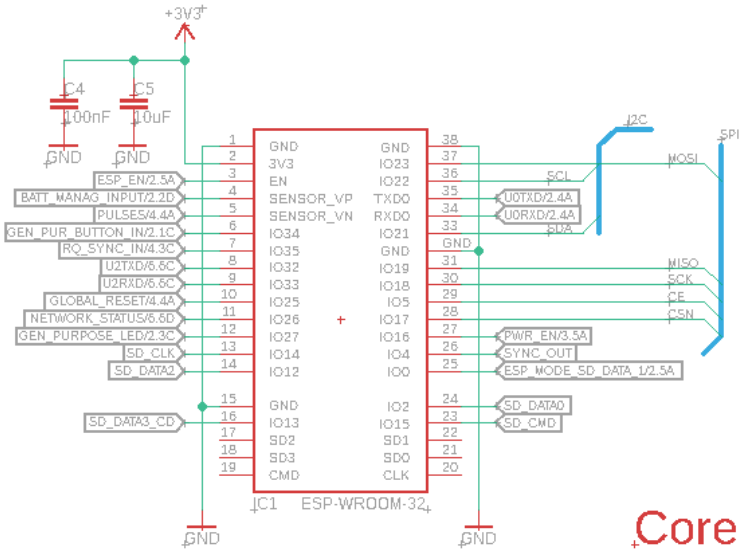
2.3 Elektrické schéma

Následující kapitola popisuje elektrické schéma Datového koncentrátoru. Kompletní elektrické schéma, na které bude v rámci této kapitoly odkazováno lze nalézt v příloze A na konci dokumentu.

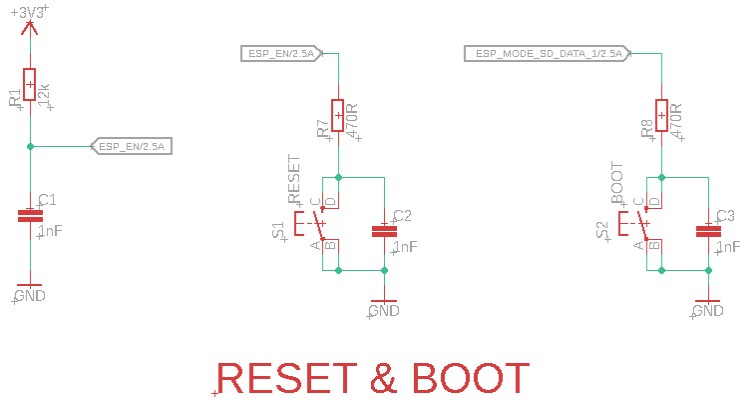
Celé schéma je nakresleno v rámci návrhového prostředí Eagle popsaného v kapitole 1.5.1 a obsahuje jak výše zmíněné a vybrané hardwarové moduly datového koncentrátoru, tak další nezbytné aktivní a pasivní prvky, které spolu po obvodové stránce vytváří kompletní zařízení.

V rámci obrázku 2.19, který je součástí prvního listu elektrického schématu si lze všimnout základního vyvedení pinů z MCU modulu ESP32-WROOM-32 (IC1). Výběr definovaných pinů pro spojení s dále diskutovanými elektrickými komponentami byl proveden na základě katalogových listů určených pro tento mikrokontrolér [18] a [19]. Dále je na obrázku 2.20 zařízení doplněno o tlačítka pro resetování (S1) a nahrávání programu (S2). Hodnoty rezistorů a kondenzátorů, stejně jako hodnoty kondenzátorů pro filtraci napájecího napětí mikrokontroléru jsou převzaty z referenčního návrhu verze 1.0 zařízení ESP32-DevKitC-V2 [20]. "Pull-up"rezistory pro SD slot J1 na obrázku 2.21 a jejich hodnoty byly použity na základě doporučení v rámci dokumentace [21] pro realizaci sběrnice pro komunikaci SD karty s mikrokontrolérem ESP32-WROOM-32. Dále jsou v rámci prvního listu vyvedení všech

použitých sběrnic na volné piny pro pozdější ladění hardwaru.



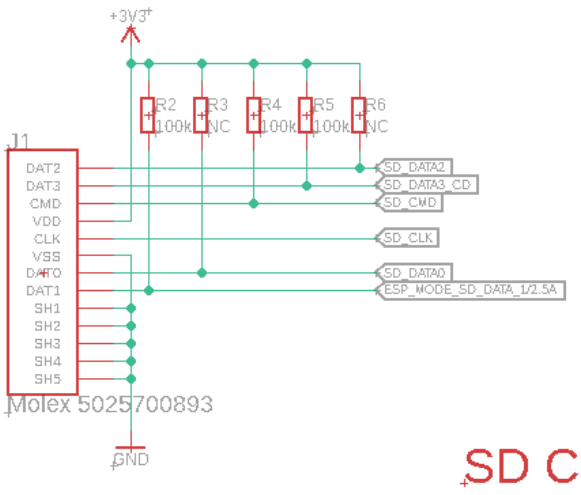
Obr. 2.19: Elektrický návrh výpočetního jádra



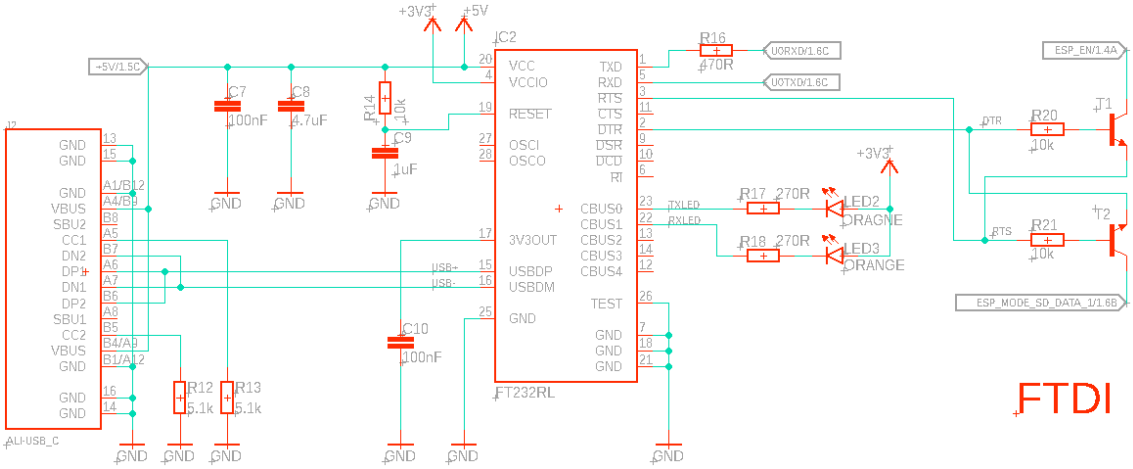
Obr. 2.20: Elektrický návrh resetovacího a bootovacího

Obrázku 2.22 z listu dva obsahuje zapojení obvodu FT232RL (IC2) s konektorem USB-C (J2). Toto zapojení bylo vytvořeno na základě dvou aplikačních nót a to "USB to MCU UART"a rozhraní "USB to RS232"z katalogového listu FT232RL [23], odkud byly převzaty všechny hodnoty kondenzátorů a rezistorů i pro indikační LED diody. Pouze část zapojení pro signály RTS a DTR je převzata z již výše zmiňovaného referenčního návrhu verze 1.0 zařízení ESP32-DevKitC-V2 [20].

Pro autoresetování obvodu FT232RL při připojení napájení byl použit doporučený odpor 10 $k\Omega$ a k němu přidán relativně velký kondenzátor o hodnotě 1 μF.



Obr. 2.21: Elektrický návrh připojení rozhraní pro SD kartu



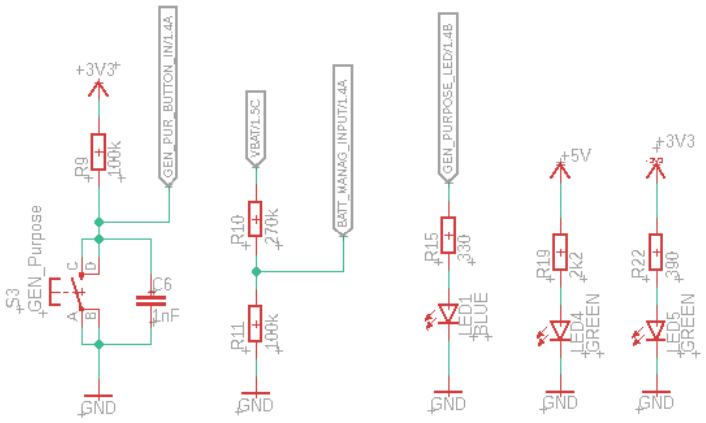
Obr. 2.22: Elektrický návrh připojení pro rozhraní USB/UART

Vzniká tak RC článek s časovou konstantou $\tau_{\mathrm{RC}} = 10\,\mathrm{ms}$, která byla vypočtena na základě rovnice 2.6.

$$\tau_{\mathrm{RC}} = R \times C = 10\,k\Omega \times 1\,\mu\mathrm{F} = 10\,\mathrm{ms} \tag{2.6}$$

Kde τ_{RC} je časová konstanta RC článku, R je hodnota odporu použitého rezistoru v RC článku a C je kapacita kondenzátoru daného RC článku.

Dále byl na základě stejného výpočtu 2.6 vytvořen RC článek pro funkci dolnofrekvenčního filtru pro potlačení zákmitů tlačítka pro obecné použití (S3) s časovou konstantou $\tau = 0,1\,\mathrm{ms}$, který lze vidět na obrázku 2.23.



Obr. 2.23: Elektrický návrh indikace stavu napájení a uživatelského rozhraní

Zde lze vidět taktéž LED diody (LED4, LED5 a LED1) jako indikátory napájecího napětí 5 V a 3,3 V a výstupní LED indikátor pro obecné použití. Volba hodnot odporů jednotlivých rezistorů pro nastavení proudu diodami byla vypočtena na základě rovnice 2.7, kdy požadovaný proud diodami je 1mA. Ze zkušeností víme, že se jedná o proud dostatečný pro zřetelný svit LED diod.

$$R_{\text{ZelenáLED}} = \frac{U_{\text{R}}}{I_{\text{ZelenáLED}}} = \frac{U_{+3\text{V3}} - U_{\text{ZelenáLED}}}{I_{\text{ZelenáLED}}} = \frac{3,3\,\text{V} - 2,9\,\text{V}}{1\,\text{mA}} = 400\,\Omega \approx 390\,\Omega \tag{2.7}$$

Kde $R_{\text{ZelenáLED}}$ je kalkulovaná hodnota odporu pro předřadný rezistor LED diody, U_{R} je úbytek napětí na tomto rezistoru, $I_{\text{ZelenáLED}}$ je požadovaný proud LED diodou, $U_{+3\text{V3}}$ je napětí napájecího zdroje a $U_{\text{ZelenáLED}}$ je úbytek napětí na diodě při proudu 1 mA.

Dále je na obrázku přítomný napěťový dělič (R10, R11) pro indikaci napěťového stavu akumulátoru, přičemž vysoké hodnoty odporů rezistorů jsou voleny tak, aby akumulátor proudově nezatěžovali a dále, aby indikované napětí dělili cirka na jednu třetinu.

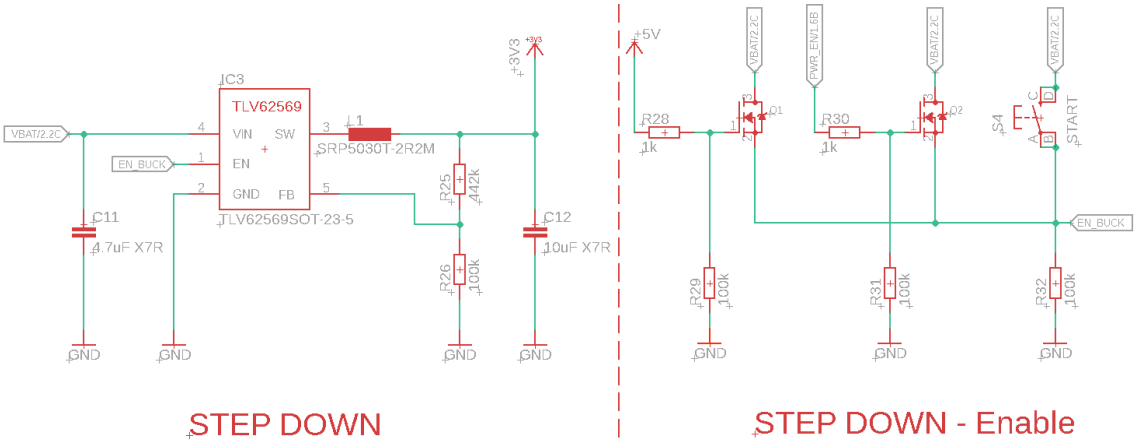
V rámci obrázku 2.24 je navržen sestupný měnič TLV62569 (IC3). Schéma je založeno na základě doporučeného zapojení dle katalogového listu [27], včetně hodnot a typů použitých kondenzátorů a cívky. Pomocí odporů R25 a R26 se pak nastavuje požadované výstupní napětí dle rovnice 2.8, 2.9 a 2.10, kdy výrobcem doporučený odpor pro R1, dále také R26 je 100 $k\Omega$.

$$U_{\text{OUT}} = U_{\text{FB}} \times \left(1 + \frac{R_1}{R_2}\right) \rightarrow R_1 = \left(\frac{U_{\text{OUT}}}{U_{\text{FB}}} - 1\right) \times R_2 \tag{2.8}$$

$$R_1 = \left(\frac{U_{\mathrm{OUT}}}{U_{\mathrm{FB}}} - 1\right) \times R_2 \rightarrow R_{25} = \left(\frac{U_{\mathrm{OUT}}}{U_{\mathrm{FB}}} - 1\right) \times R_{26} \tag{2.9}$$

$$R_{25} = \left(\frac{3,3\,\mathrm{V}}{0,\,6\mathrm{V}} - 1\right) \times 100\,k\Omega = 450\,k\Omega \approx 442\,k\Omega \tag{2.10}$$

Kde U_{OUT} je výstupní napětí měniče, U_{FB} je regulované zpětnovazební napětí a $R1\,(R25)$ a $R2\,(R26)$ jsou stavitelné hodnoty odporů jednotlivých rezistorů.

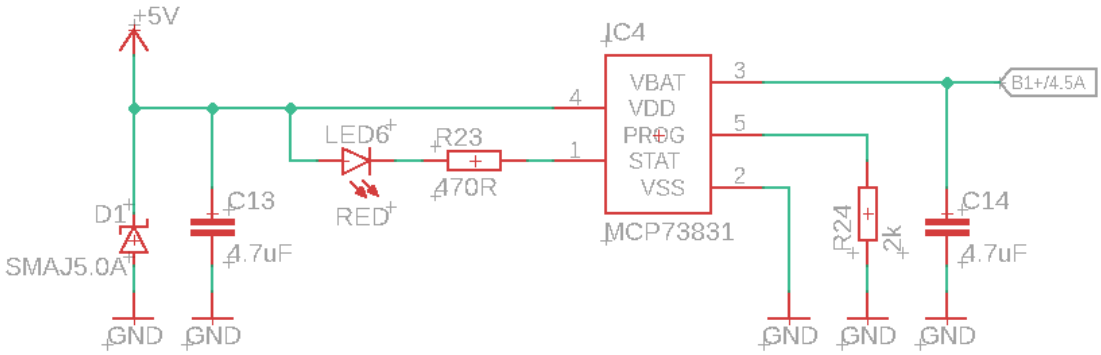


Obr. 2.24: Elektrický návrh sestupného měniče TLV62569

Na obrázku je taktéž zapojení pro povolování operace obvodu TLV62569. Zapojení bylo vytvořeno na základě principiálních znalostí tranzistorů, kdy jsou obvody odporů voleny tak, že pokud není na pinu "Gate"tranzistoru napěťový povolovací signál (5 V nebo 3,3 V), je "Gate"tranzistoru stažen k potenciálu země a tranzistor je vypnutý. Pokud je však napěťový signál pro povolení přítomen, na "Gate"pinu se v rámci napěťového děliče objeví takřka celé napětí povolovacího signálu, které tranzistor dostatečně otevře a na pinu "EN"měniče TLV62569 se objeví dostatečné napětí pro povolení jeho činnosti.

Dále v rámci listu tři na obrázku 2.25 lze nahlédnout na nabíjecí obvod pro Li-Ion akumulátor MCP73831 (IC4), kdy jeho schéma včetně hodnot komponent bylo převzato z modelového zapojení v rámci jeho katalogového listu [28].

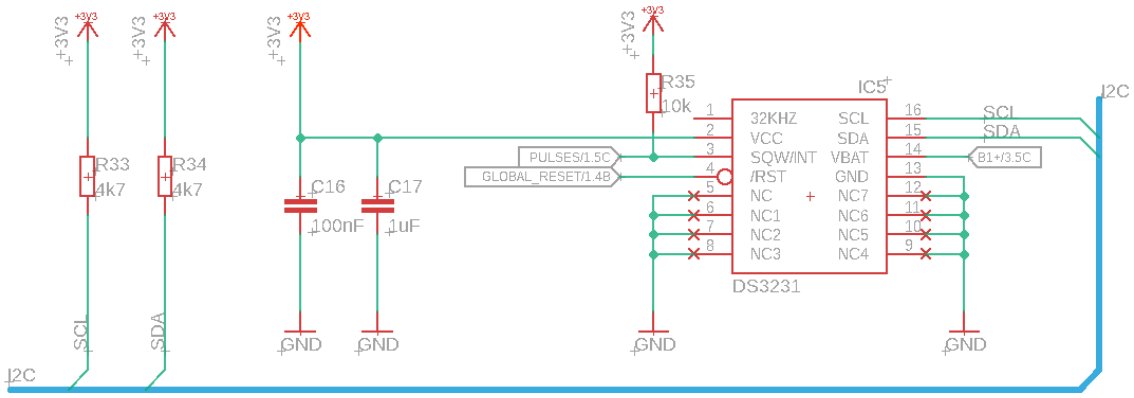
List čtyři elektrického návrhu Datového koncentrátoru popisuje na obrázku 2.26 připojení obvodu reálného času DS3231SN (IC5) k MCU modulu v rámci sběrnice I2C a dvou jednopinových spojení. Pro filtraci napájecího napětí byly opět použity pro tento účel typické hodnoty $1\,\mu$F a 1 nF. Dále hodnoty "pull-up"rezistorů (R33, R34) pro sběrnici I2C nebyly vypočteny dle katalogového listu, který pro výpočet předpokládá znalost, či odhad celkové kapacity sběrnice v zařízení, ale hodnota



Obr. 2.25: Elektrický návrh nabíjecího obvodu MCP73831

$4,7\,k\Omega$ byla použita s ohledem na to, že je základní a velmi běžná hodnota "pull-up"rezistorů použitých v praxi pro sběrnice I2C.[29]

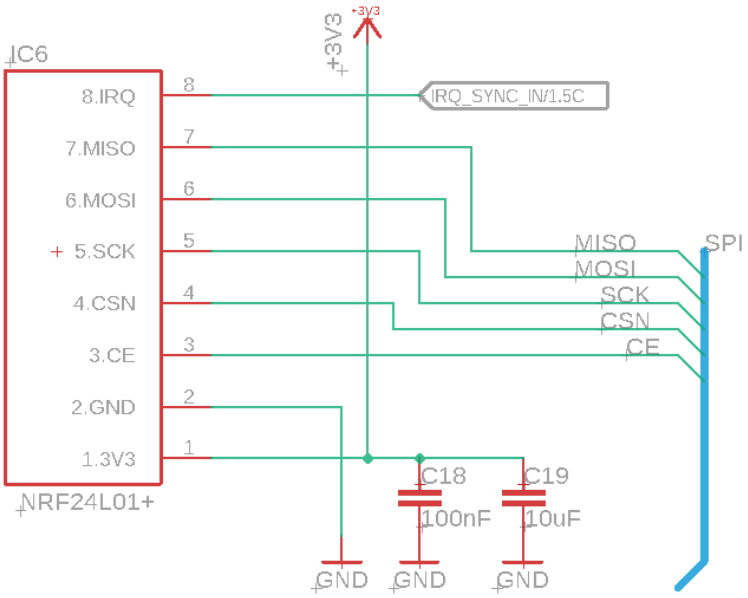
V neposlední řadě je zde k pinu INT/SQW sloužícího k budoucímu generaci přesných sekundových pulzů připojen dle katalogového listu "pull-up"rezistor (R35) s hodnotou $10\,k\Omega$. Obvod je taktéž připojen přímo k napájecímu akumulátoru pro uchování nastaveného času i při vypnutém zařízení.[17]



Obr. 2.26: Elektrický návrh pro připojení integrovaného obvodu DS3231SN

V rámci obrázku 2.27, který pochází z dolní části listu čtyři elektrického schématu je vyobrazeno připojení rádiového komunikačního modulu nRF24L01 (IC6) pomocí sběrnice SPI. Dále jsou opět použity již výše zmíněné hodnoty kondenzátorů (C18, C19) sloužící pro filtraci napájecího napětí.

Poslední dva listy elektrického návrhu, tedy listy pět a šest se zabývají elektrickým zapojením modulu SIM808 a dále jeho připojením k výpočetnímu jádru Datového koncentrátoru ESP-WROOM-32.



Obr. 2.27: Elektrický návrh pro připojení rádiového modulu nRF24L01

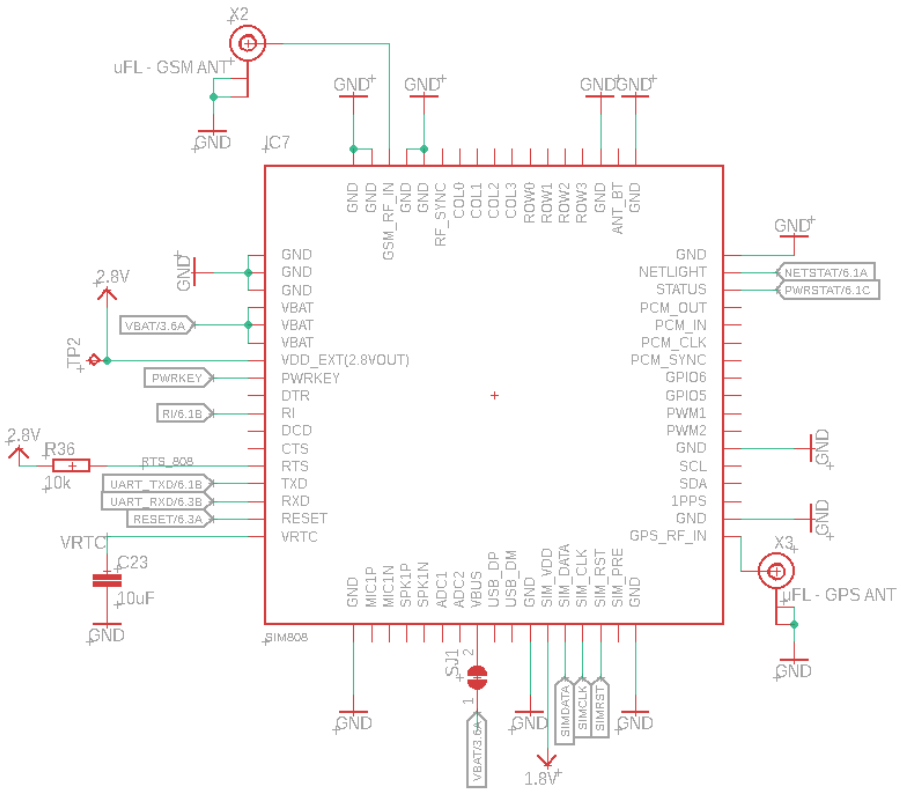
Návrh, včetně použitých komponent a jejich hodnot pro zapojení modulu SIM808 v rámci Datového koncentrátoru byl převzato z vývojové desky Adafruit FONA 808 a katalogového listu verze 1.3 určeného pro návrh hardwaru s modulem SIM808.[11, 30]

Na prvním výřezu 2.28 z listu pět je samotný modul SIM808 (IC7) s vyvedenými signály z modulu, doplňujícím zapojením a dvěma uFL konektory X2 a X3 pro připojení pasivní GPS a GSM/GPRS antény.

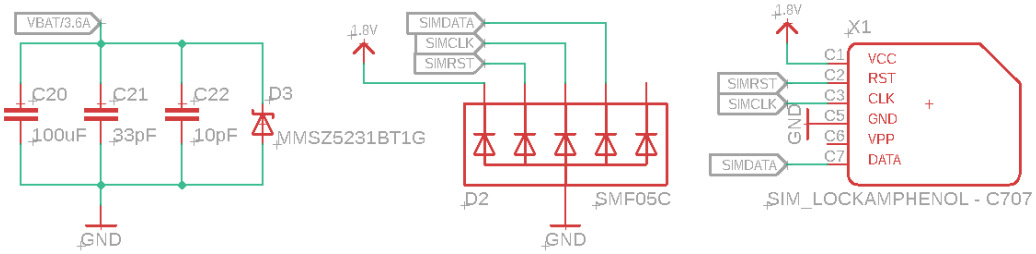
Dále na obrázku 2.29 vlevo vidíme filtraci napájecího napětí pro modul SIM808 i s přepěťovou ochranou, uprostřed a vpravo dále vidíme připojení slotu pro SIM kartu X1 k modulu SIM808 včetně ochrany proti ESD D2.

Poslední list elektrického schématu (List 6) pojednává o připojení modulu SIM808 k MCU modulu přes sběrnici UART a nutné konverzi napěťových úrovní v rámci této sběrnice. Použitím posilovače napětí 74VHC125 (IC8) se konvertuje 2,8 V logika modulu SIM808 na 3,3 V logiku MCU modulu. Dále pomocí reverzního zapojení diod s "pull-up"rezistory se konvertuje 3,3 V logika zpět na logiku 2,8 V. Na obrázku 2.30 je uveden příklad této konverze z listu šest elektrického schématu.

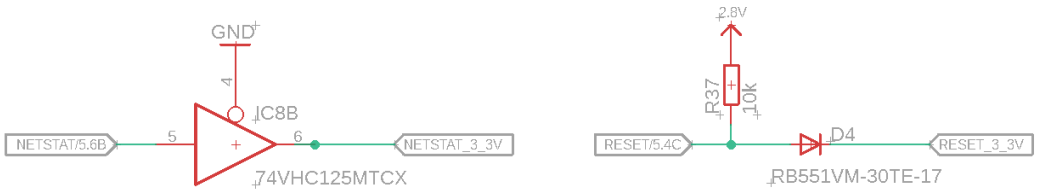
Dále je v rámci listu šest uvedeno zapojení dvou LED indikátorů (LED8 a LED9) aktuálního stavu modulu SIM808. Hodnoty rezistorů byly pro použité LED diody opět vypočítány dle rovnice 2.7.



Obr. 2.28: Elektrický návrh pro připojení modulu SIM808



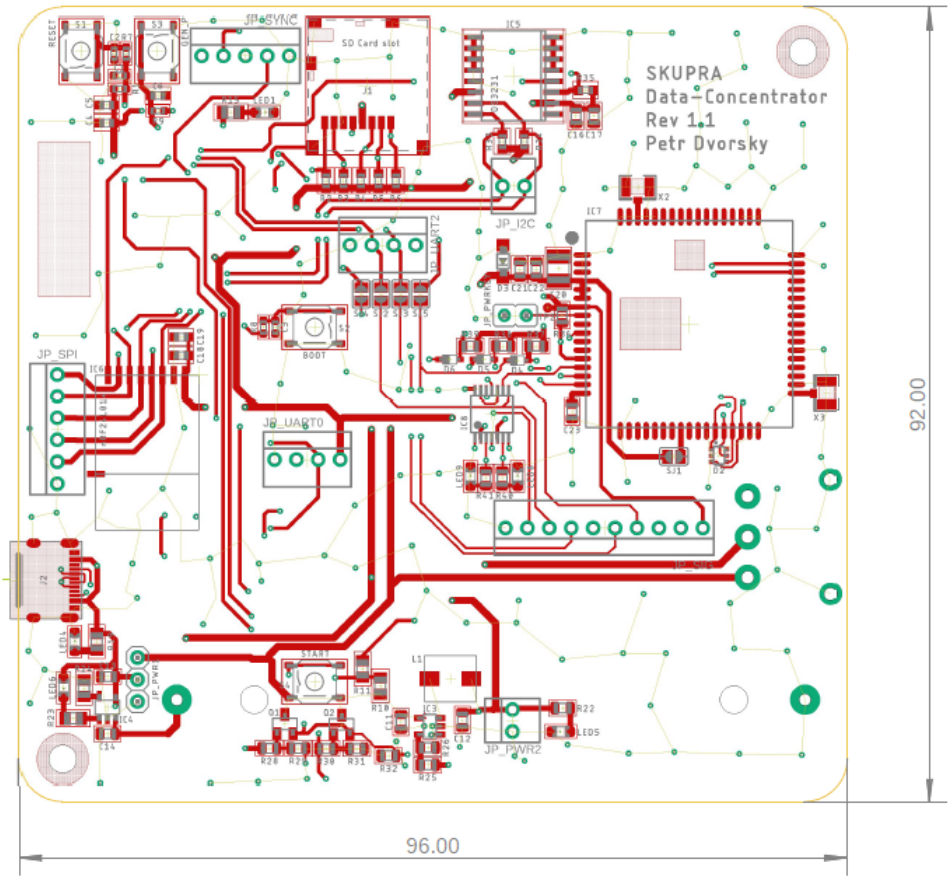
Obr. 2.29: Elektrický návrh filtrace napětí a připojení SIM slotu k SIM808



Obr. 2.30: Příklad zapojení pro konverzi dvou napěťových úrovní

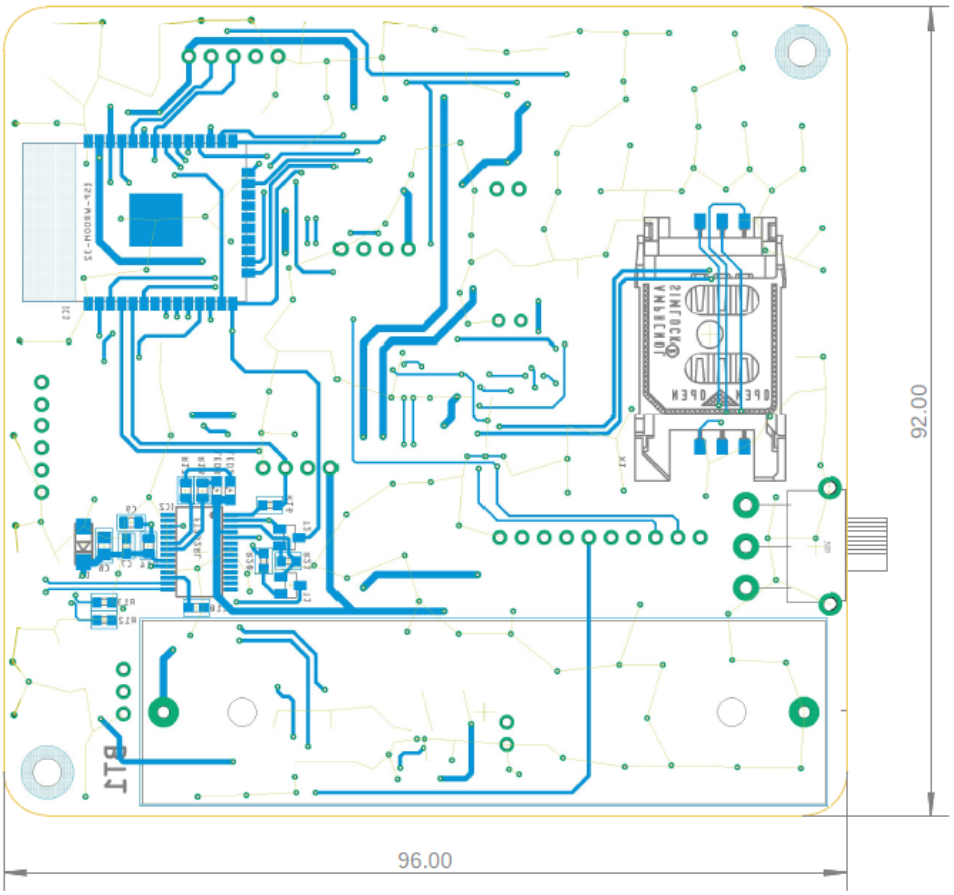
2.4 Deska plošných spojů Datového koncentrátoru

Na obrázku 2.31 a 2.32 lze nahlédnout na návrh první verze (verze 1.1) desky plošných spojů Datového koncentrátoru vytvořenou taktéž v softwarovém nástroji Eagle. Celková realizace zařízení, která je založena na tomto návrhu již byla diskutována výše v rámci kapitoly 2.2



Obr. 2.31: Přední strana DPS Datového koncentrátoru - Verze 1.1

V dolní levé části přední strany si lze všimnout USB-C konektoru J2 navazujícího přímo na integrovaný obvod FT232RL IC2 a taktéž na nabíjecí obvod pro Li-Ion akumulátor BT1. Ze zadní strany desky na obrázku 2.32 si lze také všimnout, že od konektoru J2 směrem k FT232RL jsou vedeny přímé trasy signály USB(+) a USB(-). Je to nejjednodušší metoda zajištění stejné délky signálových tras, která je pro USB komunikaci nutná. Směrem od konektor J2 dolů z horní strany desky dále nalezneme realizaci nabíjecího obvodu MCP73831 (IC4) realizaci a napravo od něj realizaci sestupného měniče napětí TLV62569 (IC3). Rozmístění komponent kolem měniče bylo navrženo na základě doporučeného rozmístění z katalogového listu [27].



Obr. 2.32: Zadní strana DPS Datového koncentrátoru - Verze 1.1

V levé horní části desky je ze spodní strany umístěn MCU modul ESP32-WROOM-32 (IC1). V blízkosti jeho boční strany, avšak z horní strany desky je přítomen rádiový modul nRF24L01 (IC6), dále ze stejné strany v horní části desky pak lze nalézt slot pro vložení SD karty J1 a integrovaný obvod reálného času DS3231SN IC5.

Na obrázku 2.31 v pravé části desky se nachází modul SIM808 (IC7) se dvěma vyvedenými uFL konektory X2 a X3 pro pasivní GSM a GPS antény. Na stejném místě pouze ze spodní strany je umístěn slot pro vložení SIM karty X1, což lze vidět z obrázku 2.32.

Dále je kolem takto rozvržených komponent realizováno drobné obvodové zapojení dle kapitoly 2.3.

Celý hardware je pak kompletován pomocí jednoduchých periferiích jako jsou LED signalizace, tlačítka a vstupně-výstupní piny pro analyzování funkčnosti realizovaného hardwaru.

3 Programové řešení

Následující kapitola se zabývá kompletním programovým vybavením Datového koncentrátoru pro jeho výpočetní jádro a jemu přidružené periferie. Kapitola se tedy podrobněji zabývá hierarchií, návrhem a tvorbou takovéhoto vybavení.

3.1 Úvod do koncepce programového vybavení

V rámci diplomové práce byl vytvořený a fyzicky realizovaný hardware vybaven kompletním programovým vybavením, které zajišťuje komplexní zprovoznění a odzkoušení funkcionalit jednotlivých výše zmíněných modulů Datového koncentrátoru. Toto vybavení tedy Datovému koncentrátoru poskytuje komplexní softwarové nástroje pro splnění na počátku vytyčených cílů, tedy zprostředkování komunikace s jednotlivými senzorickými jednotkami v rámci vlastního komunikačního protokolu, ukládání naměřených dat z připojených senzorů a taktéž ze samotného Datového koncentrátoru ve standardizovaném XML formátu na úložiště v podobě SD karty v rámci jednoduchého souborového systému a distribucí těchto získaných dat do cloudového prostředí ThingSpeak, které je podrobně rozebráno v rámci kapitoly 4.

Jak již bylo zmíněno v rámci kapitoly 1.5.3 byl software pro Datový koncentrátor, přesněji pro MCU modul ESP32-WROOM-32, vytvořen v rámci Microsoft Visual Studio Code v doplňku PlatformIO.

V tomto doplňku je MCU moduly možné programovat v rámci různých frameworků, přičemž v rámci této diplomové práce se budeme dále zabývat pro účely programování ESP32-WROOM-32 dvěma k tomu určenými frameworky, a to Arduino Frameworkem a Espressif IoT Development Frameworkem.

Výsledná realizace programového vybavení diplomové práce pro MCU modul je kompletně psána v rámci Espressif IoT Development Frameworku v jazyce C/C++. Dále je zdrojový kód zařízení založen na operačním systému reálného času FreeRTOS, který je popsán níže v kapitole 3.2 a jehož užití je podrobněji popsáno v rámci kapitoly 3.3. V rámci zdrojového kódu bylo taktéž využito velké množství knihoven, ať už vlastních, či knihoven třetích stran. Tyto knihovny budou podrobněji rozebírány níže v rámci kapitoly 3.5. Zde je taktéž důležité zmínit, že v rámci tvorby zdrojového kódu bylo nutné použít i knihovny třetích stran, které jsou nativně určeny pro Arduino Framework. Kompatibilita těchto knihoven a funkcí je v rámci ESP-IDF frameworku řešena pomocí přidáním Arduino doplňku jako komponenty zdrojového kódu. [31, 32]

Podstatné taktéž je, že všechny důležité komponenty zdrojových kódů pro Datový koncentrátor jsou patřičně komentovány v rámci dokumentačního standardu

pro nástroj Doxygen. Ten umožňuje automatické generování dokumentace zdrojového kódu na základě takto provedeného komentáře. Další nepopiratelnou výhodou tohoto standardizovaného komentáře je nativní podpora v rámci VS Code, kdy při použití jakékoliv takto popsaná komponenty je možné zobrazit si její nápovědu s komentářem a popsanými parametry přímo při tvorbě kódu.

Soubory kompletního programového vybavení pro Datový koncentrátor se nachází v příloze na CD přiloženém k této diplomové práci.

3.2 FreeRTOS

Jak už bylo zmíněno výše, zařízení využívá operační systém reálného času FreeRTOS v tomto případě je to speciálně modifikovaná verze ESP-IDF FreeRTOS určená pro vícejádrové zařízení typu ESP32.[33, 34]

ESP-IDF FreeRTOS, v našem případě dále jen FreeRTOS podporuje SMP (Symmetric Multiprocessing - Symetrický multiprocesing), což umožňuje vytvořit vícevláknový program na více jádrech, na kterých lze spouštět paralelně více úloh. Zařízení tedy umožňuje skutečný paralelismus. Na každém jádře, v toto případě na dvou, běží vlastní kopie OS s vlastním plánovačem úloh. Plánovače na sobě pracují nezávisle, avšak spouštějí úlohy ze společného sdíleného listu čekajících úloh. Pokud má být daná úloha vykonávána je takzvaně připnuta k jednomu ze dvou případně volných jader.[35]

Plánovače v rámci FreeRTOS se primárně zaměřují na prioritu jednotlivých úloh. Pokud má více úloh stejnou prioritu, pro volbu spuštěné úlohy je aplikován modifikovaný preemptivní algoritmus Round Robin. V tomto případě modifikace spočívá pouze ve faktu, že se plánovač taktéž zajímá o to, na jakém jádře může daná úloha běžet, tedy zda je daná úloha pevně spjata s některým z jader a její vykonávání tak je tímto podmíněno. Vhodné je v tom případě zmínit, že při nevhodném uživatelském přiřazení úloh jednotlivým jádrům může v krajních případech docházet k přeskakování některých úloh a jejich nevykonáváním. To je jedna ze skutečností, proč v rámci zdrojového kódu Datového koncentrátoru nebyla žádná úloha pevně spjata s konkrétním jádrem zařízení. Dále jsou však ve zdrojovém kódu možnosti využití vícero úloh v rámci zařízení pomocí FreeRTOS plně zastoupeny a využívány a jejich hierarchie a funkce jednotlivých úloh jsou dále popsány v rámci kapitoly 3.3. [34]

Dalším velice důležitým aspektem FreeRTOS a důvodem pro jeho použití je fakt, že jsou v rámci něj nativně implementovány prostředky pro komunikaci mezi jednotlivými úlohami jako například semafory a notifikace a prostředky pro přístup ke sdíleným prostředkům a do kritických sekcí kódu, které jsou ohraničeny zámky, neboli mutexy, kde v jeden čas může být kód vykonáván pouze jednou konkrétní

úlohou a kde taktéž k danému sdílenému prostředku muže přistupovat pouze jedna konkrétní úloha. [33]

Vše výše zmíněné je s výhodou využito právě v případě budování komplexnější aplikace, což Datový koncentrátor bezpochyby je, kdy je právě nutná separace jednotlivých funkčních celků zařízení pro její přehlednost ale taktéž, aby mohli pracovat ve skutečném paralelismu, v rámci jednoho jádra pak případně v režimu multitaskingu.

3.3 Hlavní program a jeho úlohy

Programové vybavení Datového koncentrátoru má podobu typické struktury většiny dnes známých programů pro vestavěná zařízení a systémy, tedy obsahuje hlavní soubor *main.cpp* a k němu přidružené přidružené soubory v rámci knihoven. V této kapitole se budeme podrobněji zabývat podobou hlavního souboru *main.cpp*, jeho strukturou a dále podrobněji jeho jednotlivými částmi.

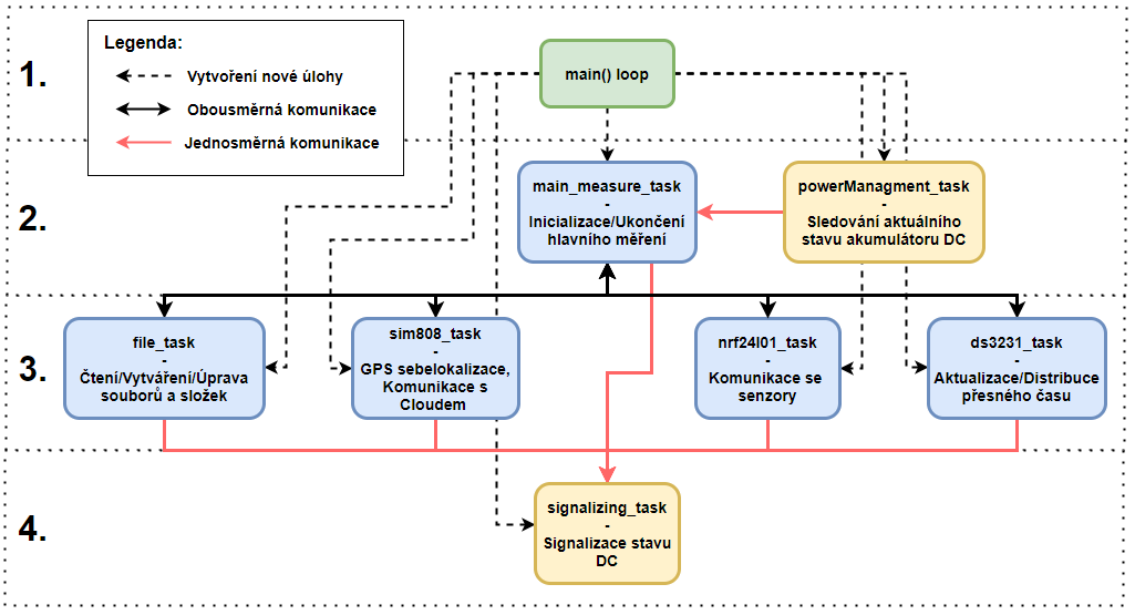
Na obr. 3.1 lze nahlédnout na uspořádání vnitřní struktury hlavního souboru. Skládá se z hlavní a první nativně volané funkce v programu, *main*() a dále z jednotlivých úloh, které jsou následně vytvořeny v rámci této funkce. Z obrázku je taktéž patrné, že jednotlivé úlohy mají hierarchické uspořádání, tedy že některé úlohy jsou jiným v jistém směru nadřazeny a povelují je. Zde je však důležité upozornit, že tato hierarchie nutně nekoresponduje s rozřazením priorit vykonávání úloh v rámci FreeRTOS. Dále jsou na obrázku naznačeny komunikační kanály mezi jednotlivými úlohami.

3.3.1 Hlavní funkce main

Tato funkce, jak již bylo naznačeno, je vždy v rámci programu volána jako první. Je tedy hlavní funkcí celého programu.

V rámci této funkce ve zdrojovém kódu Datového koncentrátoru je jako první inicializována komponenta pro kompatibilitu s Arduino knihovnami a dále zahájena sériová komunikace s případně připojeným PC pro informativní komunikaci. Tato komunikace je považována jako standardní vstup/výstup.

V dalším sledu jsou postupně nastaveny obecné vstupy a výstupy včetně přerušení, dále je provedena inicializace potřebných mutexů a binárních semaforů v rámci FreeRTOS. Jako poslední je ve funkci *main*() vytvořeno všech osm úloh, které následně převezmou kontrolu nad programem. Tímto úloha funkce main v rámci programu končí.



Obr. 3.1: Hierarchie úloh v rámci hlavního programu

3.3.2 Úloha hlavního měření

Tuto úlohu lze považovat po skončení funkce main jako hlavní, či velící úlohu. Tato úloha, celým jménem *main_measure_task* se totiž stará o průběh celého měření koncentrátoru, tedy o jeho zahájení, nastavení a ukončení. Úloha na základě konfigurace Datového koncentrátoru, o které bude diskutováno v rámci kapitoly 3.3.3 taktéž určuje, kterým podřízeným úlohám má být vlastně umožněn běh.

Přestože se jedná o sekundární hlavní úlohu, či funkci, není úloha hlavního měření nikterak složitá.

Na začátku hlavní měřicí úloha čeká na signály zdárné inicializace jí podřízených úloh $file\_task, sim808\_task, nrf24L01\_task$ a $ds3231\_task$. Po příchodu signálů úspěšné inicializace všech podřízených úloh v podobě binárních semaforů měřicí úloha nastaví nový stav "Připravena k měření"a vyčkává na povel pro spuštění měření. Signál spuštění je vygenerován při zmáčknutí tlačítka pro obecné použití, avšak tento postup bude v krátkosti zmíněn v rámci kapitoly 3.3.7 níže.

Po spuštění měření se mění stav zařízení na "Inicializace měření", provede se potřebná konfigurace aktuálního měření, jako zjištění přesného aktuálního času, připojení předkonfigurovaných měřicích uzlů (viz kapitola 3.4). Pokud je provedena inicializace bez chyb jsou zaslány signály o zahájení měření jednotlivým, v rámci měření použitým, úlohám.

Úloha následně čeká na pokyn k ukončení měření. Ten může přijít opět od tlačítka pro obecné použití nebo z úlohy *powerManagment_task* v rámci kriticky

nízkého stavu napětí na napájecím akumulátoru.

Při ukončení měření úloha opět změní svůj stav na "Ukončuji Měření"a obvolá všechny předtím inicializované úlohy signálem pro ukončení měření. Všechny úlohy ukončí činnost v rámci měření a zašlou signál jejich korektního ukončení. Hlavní úloha vymaže konfiguraci právě ukončeného měření a změní stav na "Připravena k měření". Následně tedy může být opět zahájeno nové měření a celý cyklus se v rámci úlohy hlavního měření opakuje.

Koncem této kapitoly je taktéž vhodné zmínit, že kdykoliv nějaká úloha někde čeká, je uspána a nezabírá tímto čekáním žádný výpočetní výkon. Nejedná se tedy o aktivní čekání spotřebovávající procesorový čas. V rámci celého zařízení jsou tak navrženy prakticky všechny úlohy, tedy pokud daná úloha někde čeká, tak je uspána a její probuzení a další běh je vyvolán patřičným signálem, či přerušením.

3.3.3 Úloha souborového systému se záznamem v XML

Tuto úlohu můžeme nalézt ve zdrojovém kódu pod jménem *file_task* a zaobírá se jak samotnou komunikací s SD kartou v rámci rámci již zmiňovaného SD/SDIO/MMC rozhraní, tak jednoduchým souborovým systémem vložené SD karty.

Na začátku úlohy je provedena inicializace vložené paměťové karty, zjištění jejího konkrétního typu, vyčtení její velikosti a je provedeno informativní vypsání její adresářové struktury, pokud v rámci karty již nějaká existuje.

Dále úloha ze souborové cesty `SD:/SETTINGS/config.txt` vyčte přítomný konfigurační soubor ve kterých jsou obsaženy veškeré informace o požadovaných spuštěných modulech Datového koncentrátoru v rámci měření, o počtu měřicích uzlů (viz kapitola 3.4) a jejich podrobných parametrech. Na jednoduchý formát konfiguračního souboru lze nahlédnout v příloze F. Po korektním načtení informací je ohlášen počet načtených měřicích uzlů a úloha je připravena pro případné spuštění v rámci hlavního měření.

Pokud je měření zahájeno a je v něm obsažena i úloha souborového systému, jako první úloha zajistí vytvoření nového souboru zahájeného měření, případně i nové složky pokud již není v rámci adresáře vytvořena. Celková souborová cesta má následující formát `SD:/RECORDS/YY_MM_DD/HH_MM_SS.xml`, kdy složka, kde je soubor uložen, vždy nese název aktuálního data měření a soubor čas zahájení takovéhoto měření.

Z přípony hlavního souboru měření je možné odvodit, že se jedná o formát XML a jeho vnitřní struktura je podrobně popsána níže v této podkapitole.

Dále je do hlavního souboru uložena podrobná informační hlavička popisující samotné měření a informace o měřicích uzlech obsažených v měření. Pro každý

měřicí uzel je taktéž vytvořen pomocný soubor s cestou `SD:/RECORDS/NODEX.txt`, kdy X je unikátní identifikátor daného uzlu.

Následně se v rámci pravidelných cyklů úloha dotazuje, zda v některém z uzlů jsou nové naměřené data, které je nutné uložit. Pokud jsou data v měřeném uzlu k dispozici, jsou uloženy do jeho příslušného pomocného souboru.

Při signálu ukončení aktuálního měření jsou bezpodmínečně zkontrolovány a případně uloženy nová data ve všech měřicích uzlech. Následně dochází ke kompletaci hlavního měřicího souboru, kdy všechny záznamy z jednotlivých pomocných souborů jsou překopírovány na jim určené pozice v hlavním měřicím souboru. Po kompletaci hlavního měřicího souboru je tento soubor z informativních důvodů přečten a vytisknut na standardní výstup.

Následně je pouze zaslán úloze hlavního měření signál o ukončení úlohy a úloha je opět připravena pro další měření.

Dilema časové náročnosti zápisu

Jak již bylo popsáno výše, v rámci úlohy je pro uložení jednoho měření nutné vytvořit hlavní soubor a k němu N pomocných. To se může zdát jako nelogické a pozorného čtenáře jistě napadne, proč nejsou nově zaznamenaná data ukládána přímo do hlavního souboru.

Tuto možnost vytvořené programové vybavení Datového koncentrátoru taktéž nabízí, avšak daň za pouze jeden vytvořený soubor v rámci jednoho měření je, že s přibývajícími novými záznamy koncentrovaných dat v souboru náročnost zápisu a s tím i potřebný čas pro zápis roste. Soubor obsahuje totiž přesnou strukturu a musí být zaručeno, že nový záznam bude vložen přesně na požadovanou pozici. K tomu v souboru slouží speciální značky, avšak než je značka nalezena, je nutné přečíst soubor od jeho začátku až k hledané značce.

Uvažme tedy případ, kdy máme hlavní měřicí soubor a v něm budeme ukládat data pro N uzlů. Uzly jsou seřazeny za sebou a všechny uzly ukládají nové záznamy se stejnou periodou. Pro vložení nového záznamu je vždy nutné nalézt značku příslušného uzlu v souboru.

Čas a tedy náročnost zápisu v tomto případě lineárně stoupá s každou novou dávkou záznamů pro N uzlů, přičemž pro každý uzel je směrnice tohoto lineárního nárůstu dána jeho pozicí v hlavním měřicím souboru.

Tato metoda ukládání tedy není vhodná pro měření dat z více zařízení s malou časovou periodou mezi měřicími body.

Řešením tohoto problému je právě již zmíněné vytvoření N pomocných souborů, kdy pro uložení nového záznamu není nutné složitě hledat speciální značku a nový záznam je pouze připojen na konec daného souboru. Tato operace má kontinuální,

ale hlavně velmi nízkou časovou náročnost a lze tedy ukládat záznamy z více zařízení s malou časovou periodou mezi měřicími body a po dlouhý čas. Jak již bylo zmíněno po ukončení měření, kdy je dostatečný časový prostor, jsou následně tyto záznamy z N pomocných souborů překopírovány do hlavního souboru na jejich určené pozice.

XML formát pro ukládání dat

V předchozím bylo zmíněno, že záznam z každého měření má standardní XML formát a je vytvářen podle předem dané šablony. Následující schéma na obrázku 3.2 popisuje právě strukturu této šablony, její úrovně a uspořádání pro vytvoření výsledného XML souboru. Pro názornou ukázku je v příloze G k nalezení podle této šablony vytvořený soubor ze zkušebního měření.

V rámci nejvyšší úrovně nalezneme na schématu 3.2 standardní XML hlavičku a celo-zastřešující element záznamu měření, který v sobě taktéž v souboru obsahuje přesný čas a datum začátku měření. V rámci něj nalezneme dva elementy, přičemž první z nich, hlavička, popisuje obecnou konfiguraci hlavního měřicího zařízení v tomto případě Datového koncentrátoru. Jsou zde uvedeny obecné popisné informace, ale taktéž informace o modulech zda byly v rámci měření využit, či nikoliv.

Element druhý již popisuje samotné měření, kdy v něm nalezneme element informační a N elementů pro uzly měření.

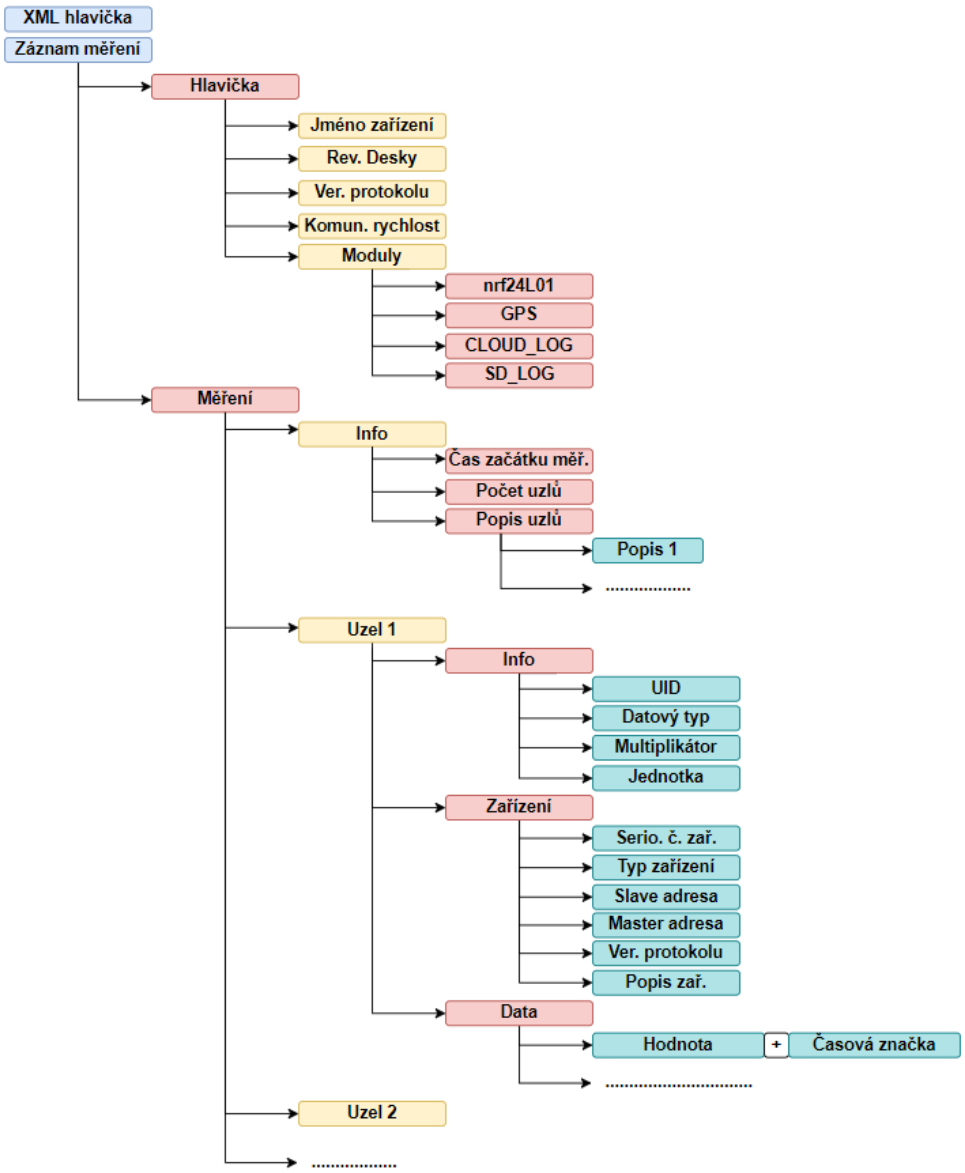
Informační element velice stručně popisuje měření jako takové, tedy datum a čas začátku měření, kolik a jaké uzly byly v rámci měření využity a jejich krátký slovní popis.

V rámci každého uzlového elementu lze nalézt, unikátní identifikátor uzlu, datový typ ukládaných hodnot, fyzikální jednotku a její multiplikátor. Dále v rámci elementu nižší vrstvy "Zařízení"detailní popis zařízení senzorické jednotky, se kterým je tento měřicí uzel spjat. V neposlední řadě uvnitř datového elementu nalezneme všechny v rámci uzlu naměřené hodnoty včetně jejich přesných časových značek. V tomto případě má časová značka formát číselné hodnota uběhnutých milisekund od počátku daného měření.

V závěru je nutné poznamenat, že popis datového typu, jednotka a multiplikátor jednotky mají v uloženém souboru podobu číselné hodnoty, neboť ve zdrojovém kódu jsou vedeny jako výčtové typy. Tabulky hodnot výčtových typů, tedy spojení čísla a významu, jsou v uvedeny přímo ve zdrojovém kódu v rámci přiloženého CD.

3.3.4 Úloha pro komunikaci se senzory

Tuto úlohu lze ve zdrojovém kódu nalézt pod celým názvem $nrf24l01\_task$. Jejím hlavním úkolem je komunikace pomocí modulu nRF24L01 přítomného na DPS Data



Obr. 3.2: Hierarchie XML souboru uloženého měření

koncentrátoru s jednotlivými senzorickými jednotkami, které jsou taktéž osazeny tímto modulem, případně čipem.

Úloha na svém začátku provede základní základní nastavení jako například nastavení komunikační rychlosti, vysílacího výkonu, či délku kontrolního součtu komunikovaných zpráv, a vypsání tohoto nastavení na standardní výstup. Následně je úloha připravena pro zahájení měření.

Pokud není úloha, a tedy i modul nRF24L01, v měření využita, je modul uveden do nízko-spotřebového úsporného režimu a úloha je v rámci aktuálního měření uspána. Pokud však v měření úloha využita je, jsou po zahájení hlavního měření

načteny adresy pro komunikaci s jednotlivými senzorickými jednotkami. To je provedeno na základě konfigurace aktuálního měření, která vychází z dříve zmiňovaného konfiguračního souboru měření v kapitole 3.3.3. Jak již bylo zmíněno v kapitole 2.1.1 pro komunikaci s každou senzorickou jednotkou, neboli zařízením, je nutné znát vždy dvě adresy, jednu na které naslouchá dané zařízení a druhou na kterém naslouchá v rámci tohoto zařízení Datový koncentrátor.

Dvojice spolu souvisejících adres vždy tvoří tkz. virtuální komunikační kanál mezi zařízeními, či virtuální tunel nebo také "pipe". Nutné je poznamenat, že modul nRF24L01 umí buď pouze vysílat a to v rámci jednoho kanálu nebo naslouchat, ale to na všech kanálech

Dále jsou tyto nastavené adresní informace vytisknuty na standardní výstup a jsou inicializovány vysílací a přijímací datové zásobníky.

V tento moment začíná Datový koncentrátor obvolávat jednotlivé senzorické jednotky. K tomu slouží pro to navržený lightweight komunikační prokol, který je založen na komunikačním modelu Leader/Follower (dříve také známý jako Master/Slave) a jeho formát bude podrobně vysvětlen v následující podkapitole.

Datový koncentrátor, v tomto případě "leader"komunikace, zašle v rámci prvního komunikačního kanálu, tedy směrem ke senzorické jednotce, v tomto případě "follower"komunikace, kontrolní zprávu o začátku měření. Pokud je dané zařízení aktivní odpoví zpět potvrzovací zprávou. Pokud ne, "leader"se pokusí o zaslání zprávy ještě dvakrát. Při úspěchu se "leader"přesune na komunikaci na dalším kanálu, při neúspěchu "leader"zkusí obvolat dané zařízení ještě dvakrát, dál se však zařízením nezaobírá a taktéž se přesune na komunikaci v rámci následně registrovaného kanálu.

Takto dojde k obvolání všech registrovaných kanálů v rámci měření a zahájení měření přímo na senzorických jednotkách. Následně v průběhu měření Datový koncentrátor naslouchá na všech kanálech příchozím datovým zprávám obsahující naměřené hodnoty, jejich časové značky a zprávy popisující metadata. Na základě zařízení, ze kterého datová zpráva přišla a metadat zprávy jsou v této zprávě naměřená data uloženy do příslušného měřicího uzlů v rámci spuštěného měření.

Při signálu pro konec aktuálního měření úloha opět obvolá všechny v rámci měření registrované kanály kontrolní zprávou pro ukončení komunikace. Princip tohoto obvolávání je totožný s výše uvedeným obvoláváním zařízení při zahájení samotného měření. Zařízení opět nazpět potvrzují zprávu o ukončení.

Po ukončení komunikace se senzorickými jednotkami úloha vyprázdní přijímací a vysílací datové zásobníky modulu, vymaže konfigurované informace o komunikaci a zašle zprávu o korektním ukončení úlohy v aktuálním měření. Úloha je dále opět připravena participovat v novém měřením.

Bezdrátový lightweight komunikační protokol

Ke komunikace mezi jednotlivými senzorickými jednotkami a koncentrátorem dochází v rámci nízkoúrovňové komunikace pomocí rádiových modulů nRF24L01, kde je již nativně implementován komunikační protokol Enhanced ShockBurst™(ESB). Ten slouží, jak již bylo vysvětleno výše v kapitole 2.1.1, pro výměnu dat mezi jednotlivými rádiovými moduly, včetně arbitrážní preambule, dynamické velikosti přenášených dat, či kontrolního součtu apod. Navržený lightweight protokol je však nadstavbou nad tímto zmíněným protokolem a definuje jak jednotnou strukturu přenášených surových dat v rámci paketu ESB (viz obr. 2.3), tak i celistvý průběh komunikace.

Lightweight protokol - Kontrolní rámec						
HEAD - Hlavička			Payload - Vlastní přenášená data			
Délka	ID	CMD	Ser. č.	Protokol Ver.	Deskriptor	Data
1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	(0-26) Bytes

Příklad -> START rámec																			
HEAD - Hlavička			Payload - Vlastní přenášená data																
						Čas začátku měření							Aktuální čas						
Délka	ID	CMD	Ser. č.	P. Ver.	Desk.	YY	MM	DD	HH	MM	SS	MS -MSB|LSB	YY	MM	DD	HH	MM	SS	MS -MSB|LSB
20	18	'D'	0x01	0x01	'T'	21	5	27	15	31	57	874	21	5	27	15	31	59	236
0x14	0x11	0x43	0x01	0x01	0x54	0x15	0x05	0x1B	0x0F	0x1F	0x39	0x03	0x6A	0x15	0x05	0x1B	0x0F	0x1F	0x3B	0x00	0xEC

Příklad --> OK rámec					
HEAD - Hlavička			Payload - Vlastní přenášená data		
Délka	ID	CMD	Ser. č.	Protokol Ver.	Deskriptor
6	15	'C'	0x01	0x01	'K'
0x06	0x0F	0x43	0x01	0x01	0x4B

Příklad --> STOP rámec					
HEAD - Hlavička			Payload - Vlastní přenášená data		
Délka	ID	CMD	Ser. č.	Protokol Ver.	Deskriptor
6	16	'C'	0x01	0x01	'S'
0x06	0x10	0x43	0x01	0x01	0x53

Příklad --> Rámec TEST KONEKTIVITY					
HEAD - Hlavička			Payload - Vlastní přenášená data		
Délka	ID	CMD	Ser. č.	Protokol Ver.	Deskriptor
6	17	'C'	0x01	0x01	'?'
0x06	0x11	0x43	0x01	0x01	0x3F

Obr. 3.3: Formát kontrolního rámce lightweight protokolu

Jak již bylo výše zmíněno, protokol je založen na komunikačním modelu Leader/Follower, kdy v roli leadera – vůdce je výše zmíněný datový koncentrátor a v roli followerů – následovníků jsou jednotlivé senzorické jednotky. Dále je v rámci protokolu definováno několik konfiguračních rámců pro možnost arbitráže (Obr. 3.3) v rámci komunikace a jeden datový rámec (Obr. 3.4) pro komunikaci již konkrétních měřených hodnot ze senzorických jednotek směrem k Datovému koncentrátoru.

Komunikaci vždy zahajuje vůdce, který nejdříve zjišťuje dostupnost jednotlivých následovníků na předem definovaných adresách. Pokud je daný následovník aktivní,

odpoví definovaným "OK"potvrzovacím rámcem. S aktivními následovníky lze následně zahájit měření speciálním startovacím rámcem, který mimo hlavičku a další parametry obsahuje i přesný aktuální čas a čas, kdy bylo samotné měření zahájeno. Následovníci tento rámec standardně potvrdí "OK"potvrzovacím rámcem a dále v průběhu aktivního měření již komunikují směrem k vůdci pouze svá naměřená data zapouzdřená ve standardním datovém rámci. Vůdce přijetí těchto datových rámců již potvrzuje pouze v rámci výše zmíněné nižší komunikační vrstvy ESB. Ukončení celého měření a komunikace opět iniciuje vůdce definovaným ukončovacím rámcem, na který následovníci standardně odpovídají potvrzovacím rámcem.

Na strukturu jednotlivých rámců a konkrétní příklady jejich podoby je možné nahlédnout na obrázcích 3.3 a 3.4. Na význam jednotlivých polí a hodnoty, která tato pole mohou nabývat lze nahlédnout v příloze E do tabulek E.1 a E.2.

Lightweight protokol - Datový rámec									
HEAD - Hlavička			Payload - Vlastní přenášená data						
Délka	ID	CMD	Ser. č.	Protokol Ver.	Datový typ	Multiplikátor	Fyz. jednotka	Měř. hodnota	Čas. značka
1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	4 Bytes	8 Bytes

Příklad -> DATA rámec																			
HEAD - Hlavička			Payload - Vlastní přenášená data																
Délka	ID	CMD	Ser. č.	P. Ver.	D. typ	Mult.	F. jedn.	Měř. hodnota - |MSB -> LSB|				Časová značka - |MSB -> LSB|							
20	18	'D'	0x05	0x01	4	0	2	2548				128 345							
0x14	0x12	0x44	0x05	0x01	0x04	0x00	0x02	0x00	0x00	0x09	0xF4	0x00	0x00	0x00	0x00	0x00	0x01	0xF5	0x59

Obr. 3.4: Formát datového rámce lightweight protokolu

V rámci komunikace je důležité poznamenat, že funkce dostupnosti jednotlivých následovníků v rámci komunikace a zahájení jejich samotného měření lze sloučit, přičemž zaslání speciálního startovacího rámce principiálně slouží i jako indikace, zda je dotyčné zařízení dostupné, či nikoliv. Tato zkrácená verze arbitráže je právě implementována ve výše popisované úloze $nrf24l01\_task$.

Navržený protokol dále zohledňuje skutečnost, že se v našem případě v rámci komunikovaných dat jedná o měření nejrůznějších fyzikálních veličin a je tedy třeba se samotnou naměřenou číselnou hodnotou taktéž přenášet její fyzikální jednotku a její multiplikátor. Dále je taktéž nutné definovat datový typ přenášené číselné hodnoty pro její korektní uložení a zpracování a v neposlední řadě je taktéž nutné definovat časovou značku této hodnoty pro její evidenci vzhledem k časové oblasti. To vše zajišťuje již výše zmiňovaný datový rámec uvedený na obrázku 3.4.

Dále si lze všimnout, že zařízení má možnost z hlavičky přijatého rámce zjistit, jakou má mít rámec délku, a tedy zkontrolovat zda rámec není nijak poškozený. Dále rámec obsahuje číselný identifikátor, podle kterého zařízení rozpozná zda se nejedná o již zastaralý, či přijatý rámec. V neposlední řadě rámec obsahuje pole pro

řídicí znak, na základě kterého může zařízení rozlišit zda se jedná o kontrolní, nebo datový rámec.

Takto postavený protokol tedy pevně definuje strukturu a průběh komunikace a dále zajišťuje, že přenášená data budou vždy bezezbytku kompletně popsána a nemůže tak dojít k jejich dezinterpretaci a nevhodnému zpracování.

3.3.5 Úloha pro komunikaci s cloudovým prostředím a sebelokalizaci

Jak už název úlohy *sim*808_*task* jedná se o úlohu, která majoritně vystavěná na funkcích modulu SIM808 a využívá jeho hardwarové a softwarové prostředky pro odesílání koncentrovaných dat do cloudového prostředí ThingSpeak, které je podrobně popsáno v rámci kapitoly 4, a pro sebelokalizaci zařízení Datového koncentrátoru pomocí GPS.

Na počátku úlohy dochází k inicializování samotného fyzického modulu SIM808 a ke spuštěním některých funkcí modulu jako například povolení komunikace pomocí GPRS pro komunikaci v rámci internetu a povolení GPS funkcionalit modulu. Po úspěšné inicializaci je následně zaslán signál hlavní měřicí úloze a taktéž úloze přesné časomíry *ds*3231_*task*.

Dále pokud je alespoň jedna ze dvou funkcionalit, sebelokalizace pomocí GPS nebo komunikace s cloudovým prostředím, v rámci hlavního měření využita je úloha při jeho zahájení měření spuštěna, v opačném případě se nastaví modul do úsporného režimu a následně je úloha uspána.

Pokud je v rámci měření úloha spuštěna a v měření je konfigurováno využito ukládání na cloud, úloha z měřicích uzlů odebere poslední, tedy nejaktuálnější naměřené hodnoty a v rámci GPRS zašle speciální HTTP dotaz směrem ke cloudovému prostředí s naměřenými hodnotami a následně zpracuje případnou odpověď. Pokud je v rámci měření využita sebelokalizace, úloha dále z modulu zjistí aktuální GPS souřadnice, nadmořskou výšku a rychlost Datového koncentrátoru. Tyto informace jsou však uloženy pouze tehdy, pokud modul nalezl dostatečné množství GPS družic a dosáhl takzvaného fixního statusu, tedy že na základě dat z družic je možné vypočíst aktuální polohu a informace je tedy možno považovat za validní.

Pokud jsou tedy zjištěna validní data jsou uloženy v rámci speciálních měřicích uzlů a pokud je konfigurováno, nahrány do cloudového prostředí. Toto úloha provádí v pravidelných intervalech po celou dobu měření.

Dále při požadavku na ukončení měření jsou, pokud je konfigurováno, poslány poslední měřená data na cloud a zaslán signál hlavní měřicí úloze o korektním ukončení úlohy *sim*808_*task* v rámci aktuálního měření. Následně úloha opět čeká na spuštění nového měření.

3.3.6 Úloha přesné časomíry

Úloha přesné časomíry *ds*3231_*task* má za úkol v zařízení nastavit a umožnit udržovat přesný čas, který je možné jednoduše distribuovat v rámci celého zařízení.

V tom případě je úloha specifická tím, že nepodléhá zcela hlavní měřicí úloze a pracuje nezávisle na ní.

Na začátku úlohy dojde je kompletnímu nastavení, v zařízení použitého integrovaného obvodu, DS3231SN, tak aby generovalo přesně jednosekundové pulzy na příslušném vstupně/výstupním pinu a v rámci výpočetního jádra, dále dojde k vytvoření a inicializací vysoce přesného časovače s rozlišením 1 μs a s opakovacím cyklem 1 ms.

Kombinace vysoce přesného časovače, pomocí kterého se čítají uběhlé milisekundy a přesných vteřinových pulzů z čipu DS3231SN, které toto čítání nulují, zajišťuje udržování aktuálního času v rámci zařízení s rozlišením 1 ms.

Dále úloha zkontroluje, zda není v čipu DS3231SN továrně nastavený čas. To se muže stát pokud zařízení ještě nikdy nebylo v provozu nebo pokud byl vytažen ze zařízení napájecí akumulátor. Pokud tomu tak je a nebylo by možné dosáhnout aktuálního času jiným způsobem, který bude dále popsán, je nastaveno předdefinované datum, které může být taktéž zcela chybné, avšak jelikož je přesně definované a známé je možné s ním dále vhodněji pracovat a chybu lépe detekovat.

Úloha následně čeká až je plně inicializován modul SIM808 zmiňovaný v předchozí kapitole. Ten má totiž v sobě nativně implementovánu možnost synchronizace s libovolným NTP (Network Time Protocol - Protokol pro synchronizaci vnitřních hodin počítačů) servem. Úloha *ds*3231_*task*, tedy na chvíli převezme hardwarové prostředky modulu SIM808 a tuto synchronizaci prostřednictvím něj uskuteční. V tomto případě se zařízení volně dostupným NTP serverem pro Českou republiku dostupném na `cz.pool.ntp.org`.

Pokud je synchronizace úspěšná a je k dispozici přesný aktuální čas, nahraje se tento čas do paměti čipu DS3231SN.

Na konec úlohy je pro kontrolu aktuální čas vytisknut na standardní výstup. Úloha je následně uspána, avšak přesný aktuální čas je možné v rámci zařízení distribuovat i nadále pomocí k tomu určených funkcí, které jsou zmíněny v rámci kapitoly 3.5. K přesnému aktuálnímu času mají tedy přístup všechny v rámci zařízení vytvořené úlohy.

3.3.7 Ostatní úlohy v rámci Datového koncentrátoru

V této podkapitole budou shrnutý další úlohy vytvořené v rámci Datového koncentrátoru, které nejsou tak komplexní jako výše zmiňované, a tedy není třeba je řadit do zvláštních kapitol.

První takovou úlohou je úloha pro kontrolu stavu napájení *powerManagment_task*, která má za úkol kontrolovat napěťový stav akumulátoru přítomného v zařízení.

Úloha pravidelně každou minutu zjišťuje napětí na napájecím akumulátoru prostřednictvím modulu SIM808, který k němu je přímo připojen a disponuje možností zjistit jeho aktuální napěťový stav s rozlišením na 1 mV a maximální chybou měření 20 mV. Úloha tedy zjišťuje, zda je na akumulátoru větší než minimální napětí, které bylo vypočtené v rámci kapitoly 2.1.7 a potřebné pro funkčnost celého zařízení. Pokud se napětí blíží této kritické dolní hranici, úloha má za úkol zajistit korektní vypnutí Datového koncentrátoru.

Pokud tedy na zařízení aktuálně probíhá měření, úloha *powerManagment_task* zašle hlavní úloze *main_measure_task* signál k ukončení měření, na základě toho dojde k normálnímu ukončení měření, jak je popsáno výše v kapitole 3.3.2. Po ukončení měření hlavní úloha měření zašle signál zpět úloze *powerManagment_task*, že je již možné zařízení bezpečně vypnout.

Následně je zařízení odpojeno od napájení. K odpojení napájení taktéž dojde v momentě, kdy na zařízení měření neprobíhá, a je tedy taktéž bezpečné je vypnout.

Dále je v rámci zdrojového kódu přítomná jednoduchá úloha pro základní signalizaci stavu Datového koncentrátoru. Jejím úkolem je řídit pouze LED signálku pro obecné použití na základě aktuálního stavu. Kdykoliv se tedy v některé jí nadřazených úloh změní stav zařízení, úloha na to adekvátně reaguje změnou světelné signalizace.

Významy světelné signalizace pomocí LED diody pro obecné použití jsou vysvětleny v tabulce 3.1 níže.

Tab. 3.1: Přehled významů LED signalizací Datového koncentrátoru

Stav zařízení	LED signalizace
Prvotní inicializace	Blikání s frekvencí 2 Hz
Zařízení připraveno k měření	Jedenkrát za 5 s třikrát probliknutí s frekvencí 2, 5 Hz
Inicializace měření	Blikání s frekvencí 0, 5 Hz
Aktivní měření	Kontinuálně svítící LED
Ukončování aktivního měření	Kontinuálně zhasnutá LED
Generální chyba zařízení	Blikání s frekvencí 20 Hz

Poslední vytvořenou úlohou v rámci koncentrátoru je úloha pro obsluhu tlačítka pro obecné použití. Pokud dojde ke zmáčknutí tlačítka dojde na příslušném vstupně/výstupního pinu MCU k vyvolání přerušení a k tomu příslušné obslužné rutiny. Ta vyvolá signál, který probudí právě tuto úlohu ze spánku. Úloha zakáže přerušení na daném pinu a čeká dokud není zmáčknuté tlačítko uvolněno. Jakmile dojde k

uvolnění je vygenerován signál pro ostatní úlohy, že došlo zmáčknutí tlačítka a je opět povoleno přerušení na daném pinu.

Úloha tedy slouží jak pro generování signalizace pro ostatní úlohy na základě tlačítka, tak i pro softwarové potlačení zákmitů na tomto tlačítku, ke kterým přirozeně dochází z konstrukčních důvodu.

3.4 Třídy a datové objekty Datového koncentrátoru

Tato kapitola je zaměřena na strukturu tříd a objektů v rámci zdrojového kódu a na jejich použití.

Na obrázku 3.5 je možné nahlédnout na zjednodušený doménový model Datového koncentrátoru. Zjednodušený proto, neboť je zde uvedena pouze struktura tříd zabývající se zpracováním a uchováváním naměřených dat a nejsou zde uvedeny další související rozhraní pro tyto třídy ani jiné třídy použité v rámci zdrojového kódu.

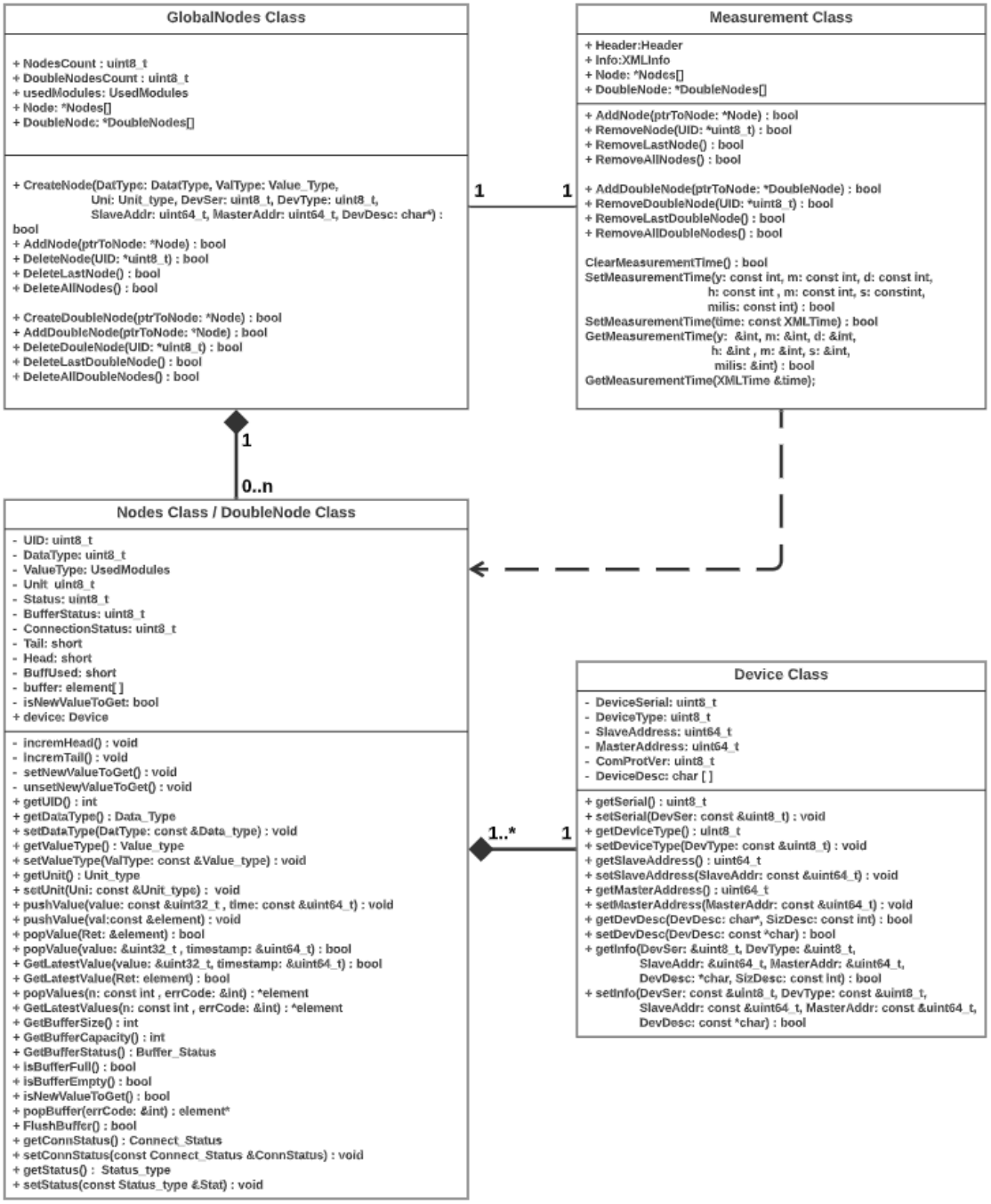
Z obrázku 3.5 lze vidět, že pro zajištění funkce měření v zařízení jsou důležité čtyři třídy.

První třída *Device class* je uzpůsobená na popis senzorické jednotky, dále také zařízení, a jejího rozhraní pro výše popsanou komunikaci skrz modul nRF24L01. Obsahuje tedy informace o adrese samotného senzoru a adrese koncentrátoru, kterou koncentrátor využívá pro komunikaci s tímto senzorem. Dále sériové číslo senzoru, definovaný tabulkový typ zařízení, verzi komunikačního protokolu a krátký slovní popis senzorické jednotky.

K ní navazující třídy *Nodes class* a speciální třída *DoubleNodes class*. Objekt vytvořený na základě jedné z těchto tříd je nazýván "Měřicí uzel", či pouze "Uzel". Každý uzel v sobě má informace o zařízení, ze kterého sbírá naměřená data, o datovém typu naměřených dat, jejich fyzikální jednotce a multiplikátoru. Dále je v každém uzlu implementován kruhový zásobník pro ukládání a distribuci spolu souvisejících dvojice hodnot a to čtyř-bajtové naměřené číselné hodnoty a osmi-bajtové hodnoty pro časovou značku. Rozdíl mezi *Nodes class* a *DoubleNodes class* je tedy pouze v tom, že v rámci *DoubleNodes class* je možné uložit dvě čtyř-bajtové naměřené číselné hodnoty místo jedné. Tento typ uzlu byl vytvořen speciálně typ dat jako GPS souřadnice, kdy jsou spolu dvě hodnoty nerozdělitelně spojeny a jedna nedává smysl bez druhé.

Jak je z doménového modelu patrné, každý tento měřicí uzel je unikátní a je spojen se zařízením. Více uzlů muže mít popis stejného zařízení, avšak v rámci jednoho uzlu v sobě uzel udržuje informace pouze o jednom zařízení.

Všechny měřicí uzly jsou vytvořeny a existují v rámci globálního objektu založeném na třídě *GlobalNodes class*. Objekt je ve zdrojovém kódu vytvořen jako globální proměnná a zajišťuje vytvoření měřicích uzlů, na základě konfigurace, včetně alokace



Obr. 3.5: UML doménový model

paměti a jejich inicializace, informace o počtu vytvořených uzlů a list s ukazateli na všechny vytvořené měřicí uzly.

Z obrázku lze vidět, že v rámci každého objektu *GlobalNodes class* lze vytvořit nula až N objektů *Nodes class* a *DoubleNodes class*. N je tu z toho důvodu, že zdrojový kód je implementován na reálné zařízení a vzhledem k paměťovým možnos-

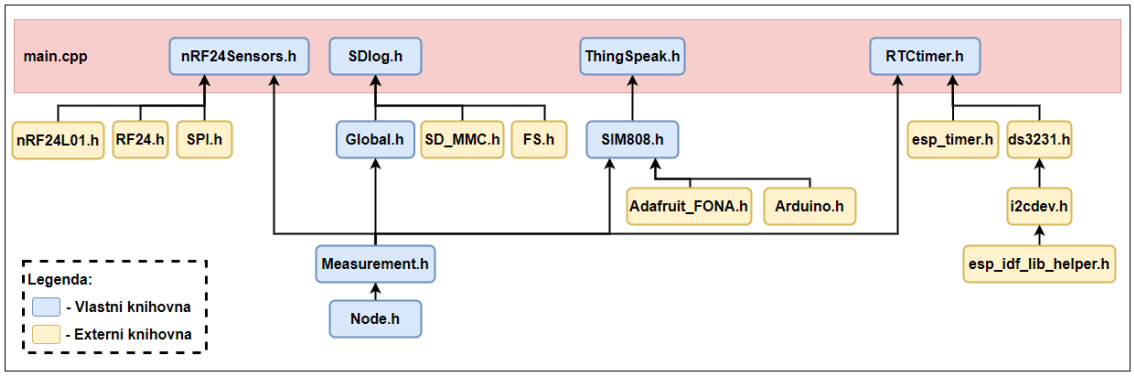
tem v něm tak nelze vytvořit nekonečně mnoho uzlů. V našem případě je N pevně nastaveno na 10.

Poslední v této kapitole diskutovanou třídou je třída *MeasurementClass*. V rámci zdrojového kódu se taktéž jedná o globální proměnou. Funkce tohoto objektu spočívá v datovém zastřešení celého aktuálního měření. V objektu jsou obsaženy všechny potřebné informace k aktuálnímu měření, tedy základní informace o zařízení, na kterém probíhá měření (Datovém koncentrátoru), čas a datum zahájeného měření, seznam použitých modulů v rámci měření, a počet měřicích uzlů a jejich seznam s odkazy do paměti, kde jsou uloženy.

Vztah mezi *GlobalNodesclass* a *MeasurementClass* je jedna ku jedné, tedy tyto objekty jsou rovnocenné a sdílející si navzájem informace. Pro objekt třídy *MeasurementClass* je tak možné získat do svého měření odkazy na uzly vytvořené na základě konfigurace a uložené v rámci objektu třídy *GlobalNodesclass*.

3.5 Knihovny pro měřicí zařízení

Tato kapitola se zabývá strukturováním použitého programového vybavení. Pojednává tedy o použitých knihovnách, jejich tvorbě a celkovou strukturovanou skladbou. Dále jsou v jednotlivých sekcích velice stručně popsány funkce a účel daných knihoven a záměr s jakými byly vytvářeny.



Obr. 3.6: Hierarchie knihoven

Na obrázku 3.6 lze nahlédnout na strukturu použitých knihoven ve zdrojovém kódu datového koncentrátoru. Z pohledu vertikální hierarchie jsou nejvýše knihovny použité v rámci hlavního souboru *main.cpp*, nejníže je potom knihovna *Node.h*, na které většina dalších knihoven staví.

O knihovnách *Node.h*, *Measurement.h* a *Global.h* již svým způsobem bylo pojednáno v předchozí kapitole 3.4. Jsou to knihovny, ve kterých jsou právě definovány

třídy z obrázku 3.5 a funkce, na kterých je postavena právě logika a distribuce naměřených dat v rámci Datového koncentrátoru.

Na obrázku z levé strany je zmíněna knihovna $nRF24L01$, ta obsahuje funkce a externí globální proměnné pro kompletní nastavení a ovládání modulu nRF24L01. Jsou zde tedy implementovány funkce od samotné komunikace přes sběrnici SPI pro nastavování registry čipu modulu až po komunikaci a zpracování dat v rámci navrženého lightweight protokolu.

Dále byla vytvořena knihovna $SDLog.h$, která v sobě obsahuje funkce pro komunikaci přes nativní sběrnici SD/SDIO/MMC, funkce pro tvorbu a ovládání jednoduchého souborové systému, funkce pro extrakci informací z konfiguračního souboru pro nastavení měření a pro generování měřicích uzlů a konečně taktéž funkce pro ukládání naměřených dat z měřicích uzlů do jednotlivých souborů včetně strukturovaného ukládání dat do hlavního měřicího souboru v definovaném XML formátu.

Další důležitou částí je knihovna $RTCtimer.h$, která v sobě, tak jak už bylo zmíněno kapitole 3.3.6, implementuje funkce a prostředky pro nastavení přesné časomíry a další distribuci aktuálního času v rámci celého zařízení.

Dále knihovna $SIM808.h$ je postavena nad knihovnami, která zaštiťují základní funkce modulu SIM808 pomocí funkcí komunikujících AT příkazy v definovaném formátu přes sběrnici UART. Knihovna $SIM808.h$ kromě kompletního nastavení modulu SIM808, slučuje tyto jednoduché funkcionality do komplexnějších funkcí. Příkladem může být funkce pro kompletní obsloužení metod POST/GET v rámci protokolu HTTP, získání aktuálního času z NTP serveru ve standardizované struktuře nebo vyčtení a zpracování informací z GPS.

Nad $SIM808.h$ je postavena knihovna $ThingSpeak.h$. Ta je navržena jako rozhraní pro Datový koncentrátor a vybrané cloudové prostředí ThingSpeak, o kterém je pojednáno v následující kapitole 4. Jsou zde vytvořeny již velmi specializované funkce pro extrakci dat z jednotlivých měřicích uzlů a jejich zasílání do cloudového prostředí ThingSpeak pomocí přesně formátovaných HTTP GET příkazů, které jsou přijímaný a zpracovány rozhraním REST API, které ThingkSpeak pro tyto účely využívá.

4 Cloudové prostředí ThingSpeak

V rámci této kapitoly bude pojednáno o nástroji ThingSpeak, který byl vybrán jako cloudové prostředí pro realizovaný Datový koncentrátor. Dále zde bude pojednáno o důvodech výběru tohoto prostředí, jeho specifikách a použití.

4.1 Výběr prostředí

V dnešní době je k dispozici velmi rozmanité spektrum dostupných cloudových prostředí, které jsou uzpůsobeny pro využití v IT světe, pro průmyslové zařízení nebo také pro IoT zařízení. Pokud se zaměříme na cloudové platformy pro IoT můžeme si vybrat z velmi širokého portfólia, ať už se jedná o nabízené prostředí od velkých korporací jako AWS IoT Platform od Amazonu, Azure IoT Suite od Microsoftu, či Google Cloud IoT Solutions. Dále je na výběr i z velkého množství menších cloudových platforem od firem, které se buď soustředí právě na provoz pouze své cloudové platformy, nebo se jedná o doplněk k jejich nabízeným produktům. Zde příkladem uvedem cloudové prostředí Thinker.io, DeviceHive, případně i Arduino Cloud.

Ve většině případů je přístup vůči uživateli takový, že je mu cloudové prostředí volně a zdarma přístupné, avšak je mu omezen výpočetní výkon, či velikost jemu vyhrazeného datového úložiště nebo má pouze omezenou paletu možností a nástrojů. V případě, že to aplikace vyžaduje, má uživatel možnost předplatit si potřebné funkce a výpočetní výkon typicky pomocí měsíčního paušálu.

Ne jinak je tomu u vybraného cloudového prostředí ThingSpeak od firmy MathWorks. Ten nabízí několik profilů v rámci platformy od základní volné verze na které je vystavěna aplikace pro Datový koncentrátor až po výpočetně prakticky neomezenou verzi standard. Detaily základní použité verze v rámci aplikace pro Datový koncentrátor budou diskutovány později.

Platforma ThingSpeak byla vybrána z ostatních možností především na základě dále popisovaných vlastností, které ji právě odlišují od jiných platforem.

Prvním a nejdůležitějším faktem je to, že se jedná o platformu, kterou poskytuje firma MathWorks, která má ve svém portfóliu taktéž produkty jako Matlab a Simulink. A právě na filozofii prostředí Matlab je tato platforma vystavěna, neboť je primárně určena pro sběr dat do cloudu a jejich další pokročilou analýzu pomocí Matlabu a pro něj určených skriptů. To přesně odpovídá aplikaci Datového koncentrátoru, kdy je požadavek na zasílání naměřených dat do cloudového prostředí, jejich vizualizaci a případné další zpracování. [36, 37]

V aplikaci je dále možno vytvářet mimo výše zmíněných výpočetních skriptů i vizualizace nebo vlastní pluginy (zásuvné mouduly) integrované do prostředí. Jeden

takový plugin založený na Google Maps, byl vyvinut a integrován pro sledování Datového koncentrátoru v rámci měření přímo na mapě v reálném čase. Dále platforma disponuje REST API (Application Programming Interface - Programové rozhraní aplikace, Representational State Transfer - Softwarová architektura API) pro velmi jednoduchou komunikaci s cloudem.

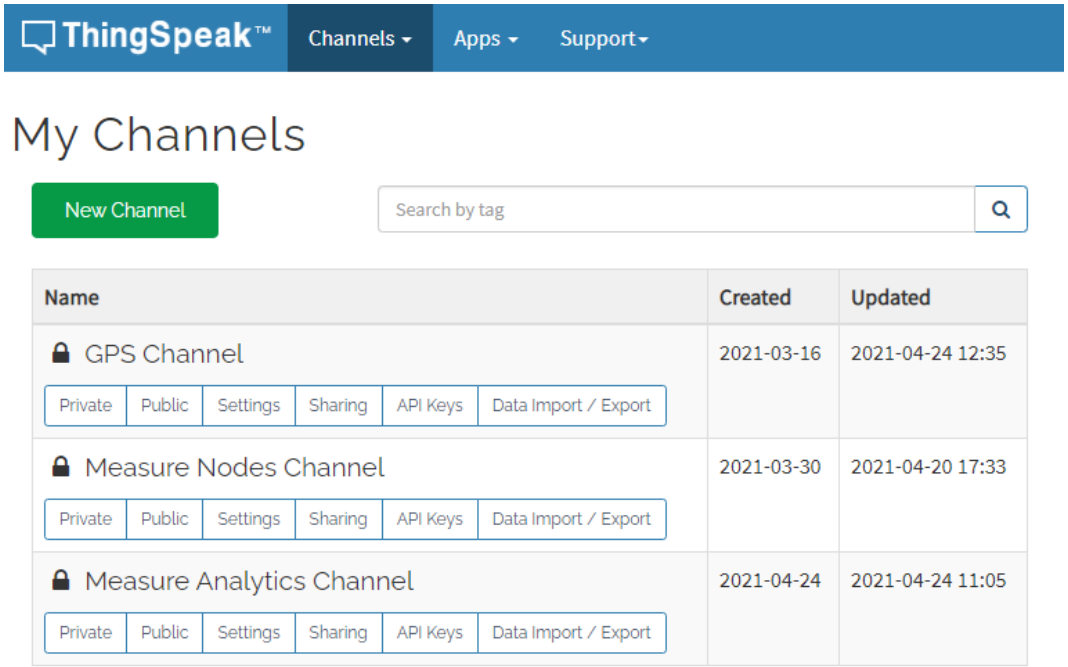
V neposlední řadě přispívá k důvěře v udržitelnosti takovéhoto cloudového prostředí fakt, že jej zaštiťuje právě velká společnost MathWorks s dlouholetou historií.

4.2 Prostředí ThingSpeak

Uživatel přistupuje do předchystaného prostředí ThingSpeak pomocí webového prohlížeče a v rámci platformy vytvořeného osobního účtu.

Prostředí lze rozdělit na dva hlavní segmenty, přičemž první je nazýván "Channels"a obsahuje v sobě všechny měřicí kanály, druhý potom "Apps", což je prostředí pro tvorbu a uložení vlastních skriptů pro vizualizaci a analýzu dat a dále pro tvorbu a úpravu vlastních, či předchystaných softwarových komponent.

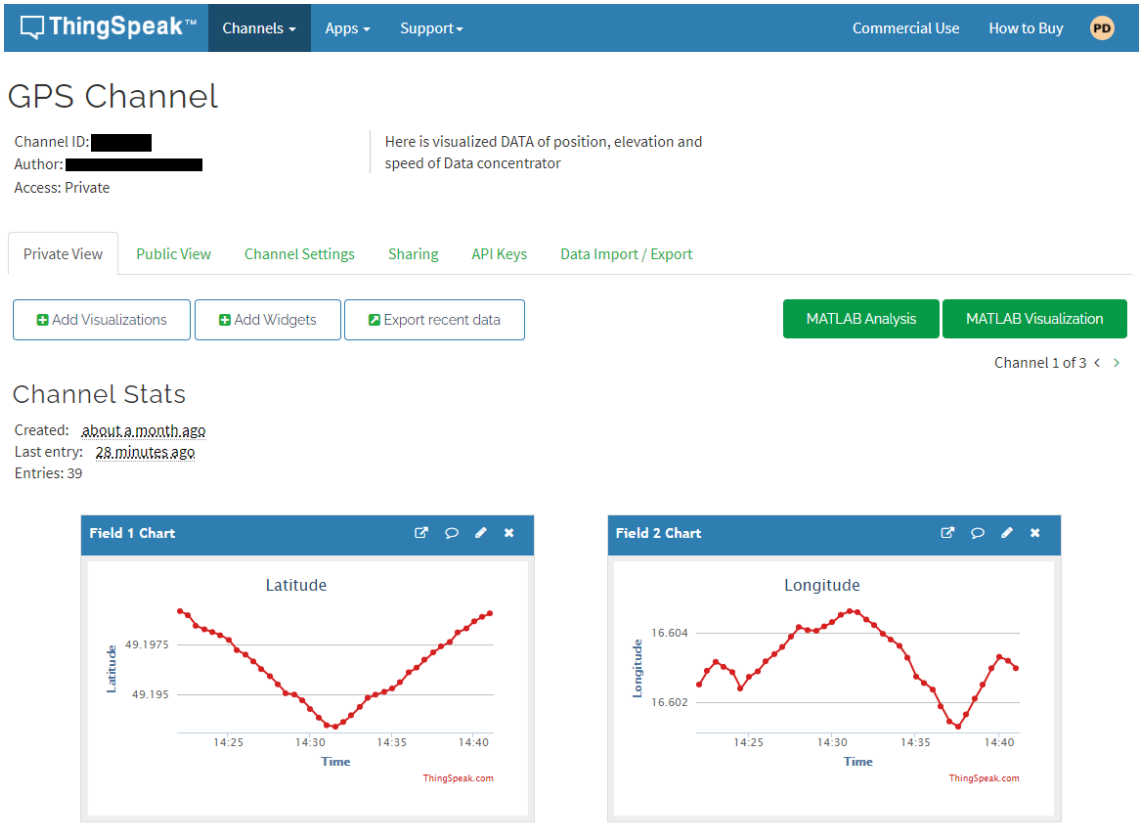
Měřicí kanály se potom dělí do sekcí vlastní, sledovaný a veřejný. Veřejný a sledovaný nás v tomto kontextu nebudou zajímat, neboť obsahují především možnosti pro uživatele, kteří chtějí svá data sdílet veřejně, případně sledovat veřejně přístupná měření.



Obr. 4.1: ThingSpeak - Vlastní měřicí kanály

V rámci vlastních kanálů (viz obr. 4.1), jsou již umístěny osobní měřicí kanály, kde je možné vytvořit nový kanál, odstranit jej, případně prokliknout se dále na jeho detail. V tomto případě, kdy je použita volně dostupná neplacená verze cloudového prostředí je možno vytvořit maximálně čtyři měřicí kanály, což je jedno z toho vyplývajících omezení. Každý kanál obsahuje maximálně osm měřicích polí, tedy oblastí pro ukládaní jedné měřené veličiny.

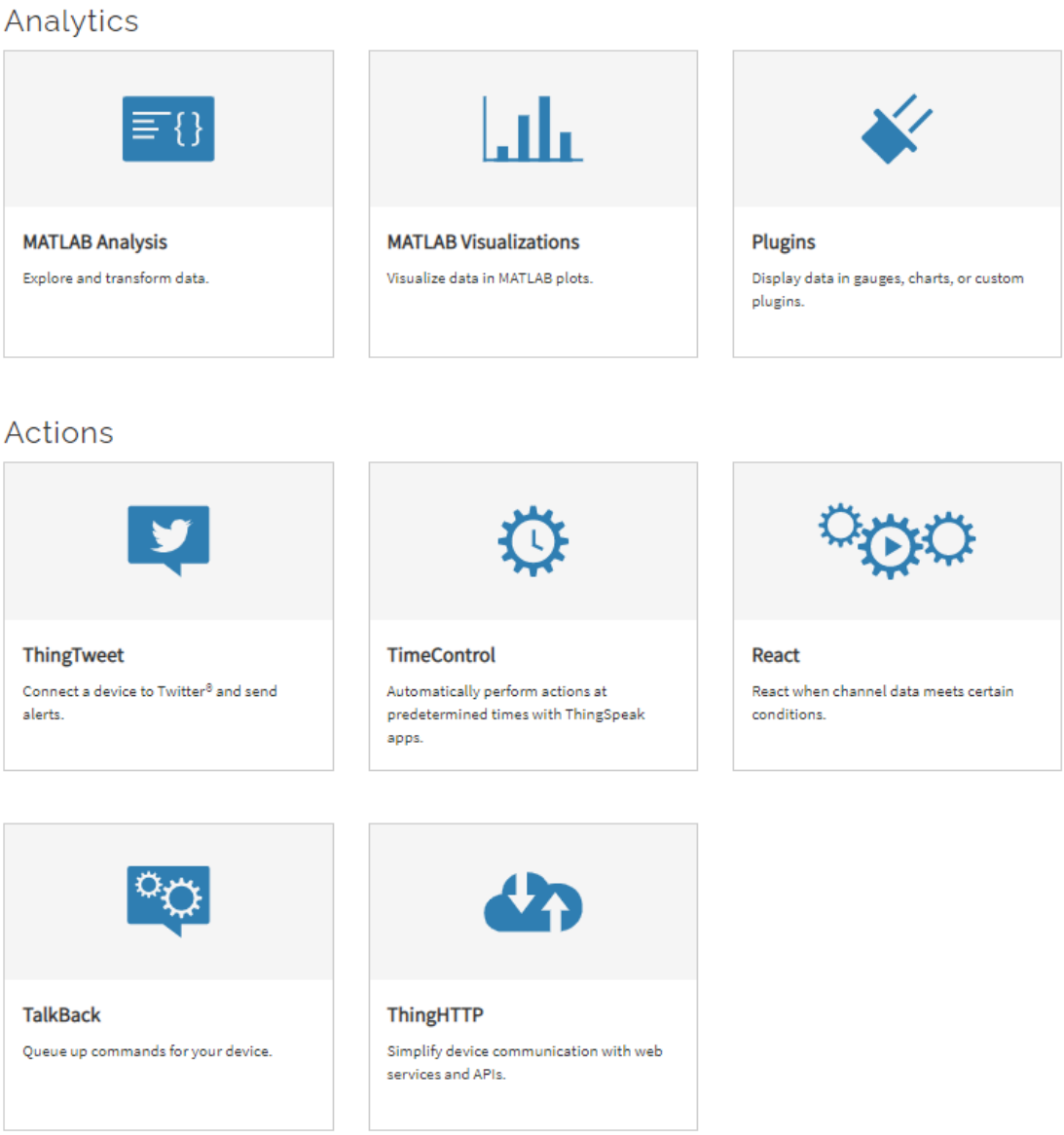
Na obrázku 4.2 je dále možno nahlédnout právě na prostředí konkrétního měřicího kanálu i s některými naměřenými daty. Je zde přítomen jeho unikátní identifikátor, identifikátor uživatele a krátký popis daného měřicího kanálu. Dále jsou zde přítomny odkazy pro podrobnější nastavení kanálu, přidání různých grafů a jiných komponent které budou zmíněny dále a generování API klíčů pro měřicí kanál. ThingSpeak taktéž umožňuje import dat do kanálu nebo export dat z něj ve standardních formátech JSON, XML a CSV.



Obr. 4.2: ThingSpeak - Měřicí kanál GPS

Jak již bylo zmíněno výše segment "Apps"je prostředí ve kterém jsou uloženy a upravovány softwarové komponenty, které jdou spojit s jednotlivými měřicími kanály. Ty pak vykonávají definované operace nad určenými nahranými daty.

Na obrázku 4.3 je výčet všech možných komponent, které mohou být využity. V rámci kapitoly 4.4 pak budou podrobně diskutovány komponenty, které byly využity nebo jsou velmi vhodné pro aplikaci Datového koncentrátoru.



Obr. 4.3: Výběr softwarových komponent pro operaci nad daty

4.3 REST API

Platforma ThingSpeak disponuje pro komunikaci s cloudem dvěma typy API a to MQTT API a REST API. [38]

Jak již bylo zmíněno výše, v tomto případě ke komunikaci naměřených dat do cloudového prostředí slouží je použito v rámci ThingSpeaku REST API.

REST je jedna ze softwarový architektur API a je postavená na modelu požadavek-odpověď. Architektura REST je nejčastěji využívaná při komunikaci v rámci protokolu HTTP.[39]

ThingSpeak používá volání REST API a jeho následující metody GET, POST, PUT a DELETE. Ty slouží k vytváření a mazání jednotlivých datových kanálů, dále pro čtení dat z jednotlivých kanálů a zápisu dat do těchto kanálů nebo také pro případné mazání uložených dat v kanálu. Komunikace s REST API a ThingSpeak cloudem, ale také serverem probíhá následovně. Webový prohlížeč nebo klient (Datový koncentrátor) odešle požadavek na server. Server následně odpoví daty v požadovaném formátu. Typicky toto rozhraní používají webové prohlížeče pro načtení webových stránek nebo k odesílání dat na vzdálené servery.[39]

ThingSpeak se však může taktéž chovat jako klient a REST API může být využito ke komunikaci s dalšími webovými servery a zařízeními, dokonce i sociálními sítěmi.[39]

V rámci tohoto zařízení je REST API využito pro nahrávání naměřených dat do cloudového prostředí ThingSpeak. K tomu slouží dvě metody POST a GET. V tomto případě jsou plně zaměnitelné, avšak obě využívají ke komunikaci dat jiný formát.

V rámci zařízení byla využita metoda GET REST API, kdy pro korektní komunikaci dat do cloudu musí být URL pro metodu v následujícím formátu:

`api.thingspeak.com/update.<format>?api_key=<api_key>&field1=<val1>`

Část URL `/update.<format>` definuje v jakém formátu má server odeslat datovou odpovědět v rámci korektního dotazu klienta. Pro odpověď ve formátu JSON je nutné vložit do URL `/update.json`, pro odpověď v XML `/update.xml`. Pokud formát není definován, tedy následovně url/update je vrácena pouze číselná hodnota opisující o kolikáté v pořadí nahrání do kanálu se jedná.

Dále je nutné obsáhnout v URL speciální ThingSpeakem vygenerovaný API klíč. Jedná se o 16ti znakový unikátní řetězec, na základě kterého platforma rozhoduje s jakým konkrétního měřicím kanálem na jakém účtu se komunikuje a zda jde o čtení, či zápis dat.

Nakonec se do URL vkládají již konkrétní data v číselné podobě. Každý měřicí kanál obsahuje maximálně osm polí, tedy v URL může být zastoupeno maximálně osm číselných hodnot ve formátu dle následujícího příkladu: `&field1=49.52&field2=15&field3=...`

V případě dat komunikovaných pomocí HTTP protokolu je nutné se ptát, zda je tato nezašifrovaná komunikace vhodná pro měřicí zařízení tohoto typu. Odpovědí může býti fakt, že pokud se externí element dostane nějakým způsobem k těmto

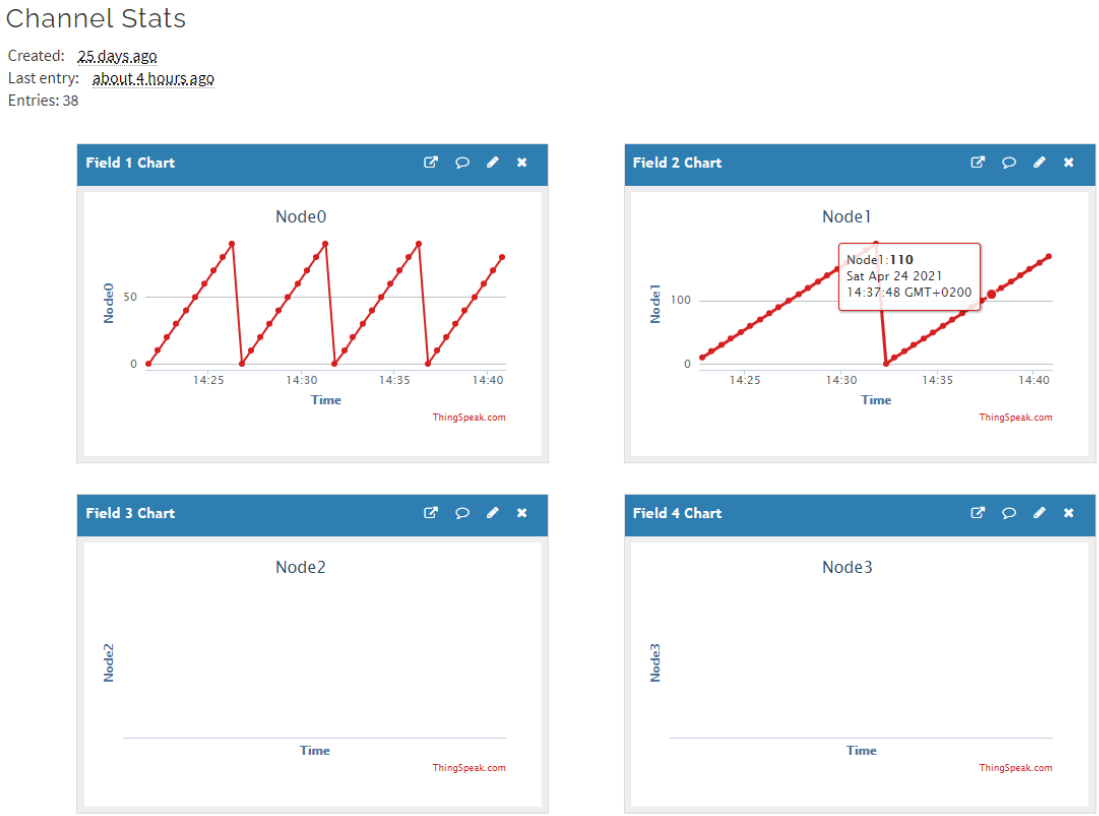
datům, tedy k řetězci obsahující URL a případné odpovědi, stejně z nich nebude mít prakticky žádný užitek neboť nemá žádné informace o tom, k čemu tyto číselné hodnoty slouží, tedy jaký je jejich zdroj a samotný význam.

4.4 Použité měřicí kanály

V této sekci jsou popsány využité měřicí kanály v rámci platformy ThingSpeak a je zde diskutována jejich konkrétní podoba a v nich využité softwarové komponenty.

4.4.1 Kanál měřicích uzlů

Nejdůležitějším vytvořeným kanálem v rámci aplikace Datového koncentrátoru je měřicí kanál "Measure Nodes Channel"(viz obr. 4.1 a 4.4). Tento kanál je určen pro ukládání a primární vizualizaci naměřených dat jednotlivých měřicích uzlů Datového koncentrátoru. V rámci takto definovaného rozhraní je možno ukládat na cloud prozatím data z maximálně osmi měřicích uzlů, to však neznamená, že aplikace nemůže býti posléze rozšířena tak, aby zpracovávala uzlů více.



Obr. 4.4: Vizualizace surových dat v kanále měřicích uzlů

Jak již bylo zmíněno výše, pro nahrávání dat je využito REST API. Uzly jsou v měřicím kanále seřazeny za sebou podle jejich UID generovaného na základě konfiguračního souboru v rámci Datového koncentrátoru. Do prvního pole měřicího kanálu jsou tedy nahrávány hodnoty z měřicího uzlu s UID = 0 až dále po případný měřicí uzel s UID = 7, který odpovídá osmému poli. Toto řazení měřicích uzlů tedy odpovídá definici těchto uzlů v konfiguračním souboru a taktéž ve výstupním souboru z provedeného měření. V rámci konfiguračního souboru je tedy možné libovolně definovat pořadí měřicích uzlů a v případě více než osmi uzlů zvolit, které měřicí uzly budou na cloud nahrávány.

Na seřazení uzlů v kanále "Measure Nodes Channel"a základní surovou vizualizaci jejich nahraných dat je možné nahlédnout na obrázku 4.4.

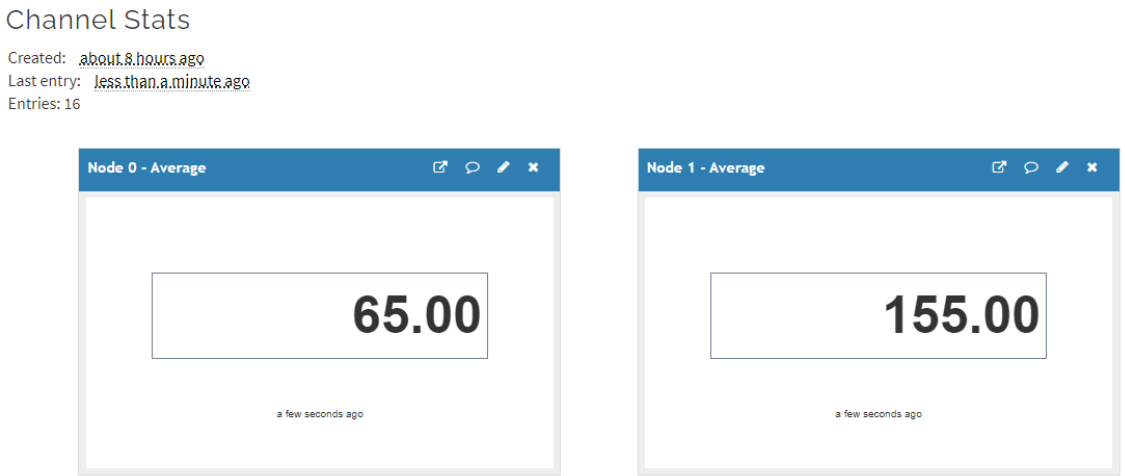
4.4.2 Zpracovatelský kanál

Pro lepší přehlednost a orientaci ve vizualizacích a dále pro případné další zpracování dat z měřicích uzlů v rámci kanálu "Measure Nodes Channel"včetně uložení nově ze skriptů vypočtených hodnot byl vytvořen kanál "Measure Analytics Channel".

Tento kanál je tedy určen pro budoucí ukládání a vizualizaci dat z vytvořených výpočetních skriptů pro Matlab, které operují nad daty z kanálu měřicích uzlů.

Pro ukázku je zde prozatím vytvořen pouze výpočet a vizualizace průměrování posledních čtyř hodnot z každého měřicího uzlu, což lze vidět v rámci obrázku 4.5. Výpočetní skript, který toto vykonává je přiložen v příloze H.

Zde je vhodné podotknou, že pro skripty a v další kapitole zmíněné softwarové komponenty v rámci ThingSpeaku přistupují k nahraným datům v kanálech taktéž pomocí REST API.

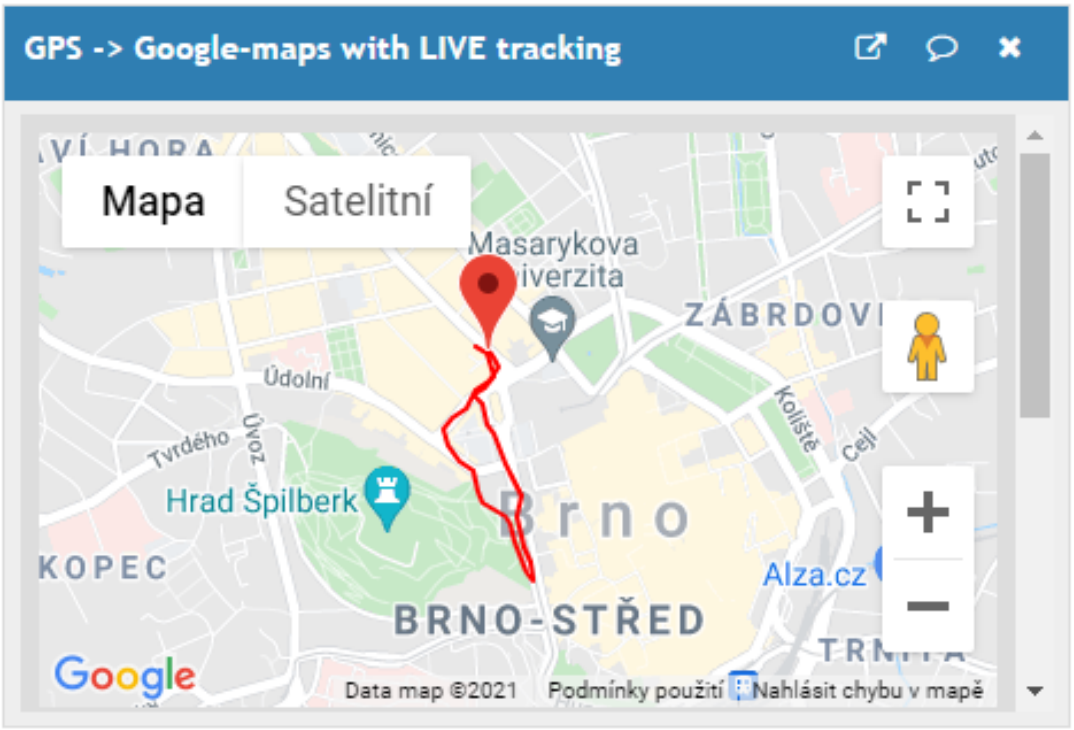


Obr. 4.5: Vizualizace surových dat v kanále měřicích uzlů

4.4.3 Kanál GPS polohy

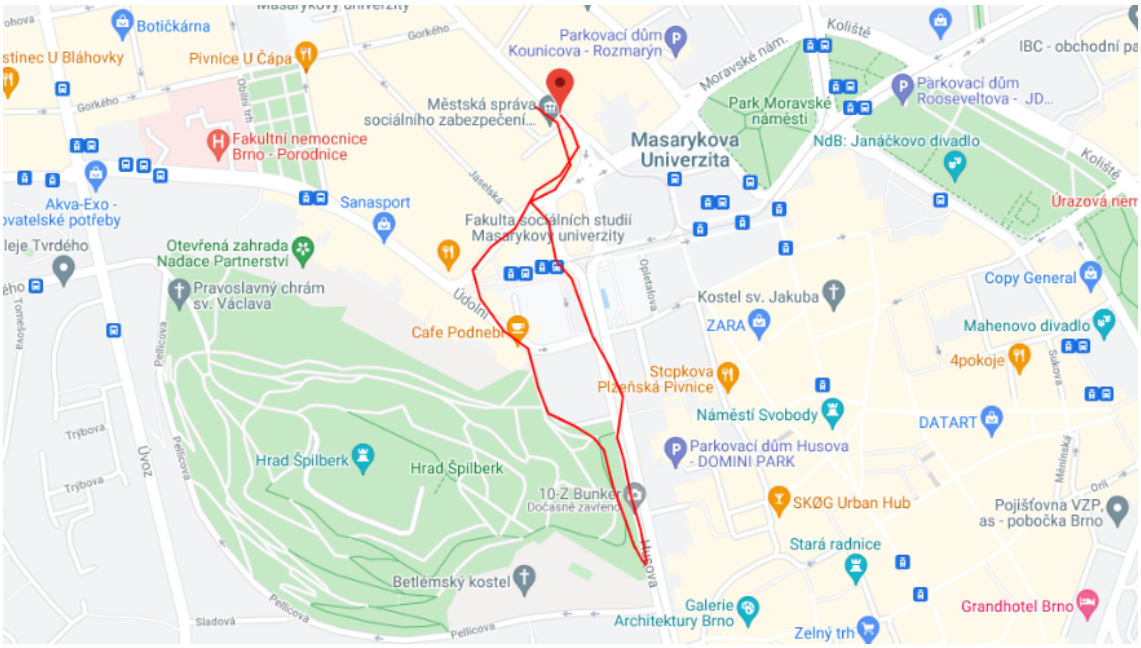
Posledním zde zmíněným měřicím kanálem je "GPS Channel"vytvořený pro účely lokalizace zařízení Datového koncentrátoru v reálném čase. Do kanálu jsou průběžně nahrávány aktuální GPS souřadnice zařízení, rychlost v jednotkách [km/h] a nadmořská výška v jednotce [m].

Pro sledování zařízení v reálném čase bylo využito možnosti vytvářet v rámci platformy ThingSpeak vlastní pluginy, přičemž pro tento účel byl vytvořen speciální Google plugin pomocí dostupných nástrojů pro vývojáře od této společnosti. Díky těmto nástrojům se podařilo integrovat do platformy modul, který v sobě obsahuje aktuální "Google Maps"se všemi jejími možnostmi a na základě naměřených souřadnic se na ni vykresluje trasa Datového koncentrátoru, kterou po čas vlastního měření urazil. Dále je v mapě vždy vyznačena oranžovou značkou aktuální poloha Datového koncentrátoru. Modul se skládá ze tří proto vytvořených souvisejících souborů v jazycích HTML, CSS a JavaScript. Na ty je možné nahlédnout v rámci přílohy na dodaném CD. Na realizovaný modul lze nahlédnout na obrázku 4.6, přiblížení trasy, kterou si uživatel může rozkliknout pak na obrázku 4.7.



Obr. 4.6: GPS plugin v prostředí ThingSpeak

Na základě výše zmíněného lze shrnout, že ThingSpeak je velmi vhodná a přímo určená platforma pro takováto různá fyzikální měření s Datovým koncentrátorem



Obr. 4.7: Uživatelské přiblížení trasy Datového koncentrátoru

s následným zpracováním, kontextualizací a vizualizací. Dále pro je tedy možné pro specializovaná měření v budoucnu prováděná na Datovém koncentrátoru přímo připravit výpočetní skripty pro zpracováním před měřením a v průběhu měření již získávat přímo požadované výsledky bez nutnosti separátní analýzy až po konci prováděného měření.

5 Zhodnocení výsledků Diplomové práce

V rámci diplomové práce jsem vytvořil Datový koncentrátor založený na mnou navržené koncepci o které bylo pojednáno v kapitole 1.1.

Na základě koncepce a výběru vhodných komponent jsem provedl elektrický návrh zařízení a vytvořil pro něj kompletní návrh desky plošných spojů. Tento svůj návrh desky plošných spojů jsem si nechal vyhotovit u externího dodavatele, který již částečně osadil desku plošných spojů některými komponenty. Zbývající komponenty jsem na desku doplnil a osadil sám. Takto vznikly celkem tři kusy prototypu datového koncentrátoru revize 1.1 (viz obr. 2.17 a 2.18).

V průběhu procesu osazování komponent došlo k odhalení první chyby návrhu desky, která je však pro funkčnost zařízení nepodstatná. Jedná o slot pro zasouvání SD karty, který je v návrhu chybně otočen o 180°.

Následně jsem vytvořeným prototypům dodal základní programové vybavení za účelem vyzkoušení realizovaného hardwaru. Provedl jsem odzkoušení výpočetního jádra a k němu všech připojených periferií a jejich funkcionalit. Soubory programového vybavení pro odzkoušení funkčnosti hardwaru je součástí přílohy na přiloženém CD.

V rámci tohoto procesu bylo detekováno několik drobných poruch, které vznikly v průběhu manuálního osazení desky plošných spojů. Typickým příkladem nefunkčnosti určité části zařízení byl nežádoucí, okem prakticky neviditelný, "cínový vlas"vniklý při ručním pájení a propojující dva sousední kontakty. Tyto poruchy byly však na základě kontrolního měření odhaleny a odstraněny.

Celé zařízení tak bylo po elektrické stránce zprovozněno a potvrdila se korektnost mého elektrického návrh zařízení.

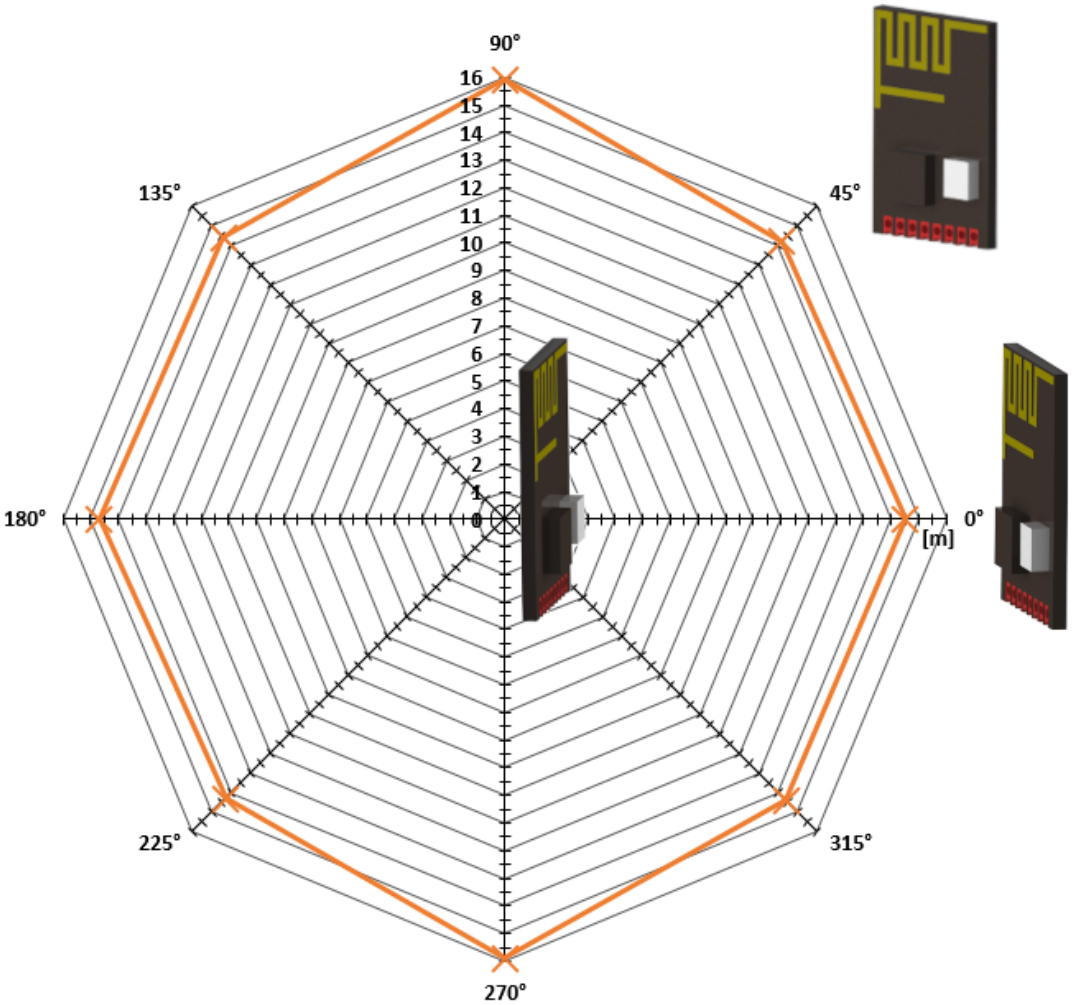
Jediný závažnější problém byl detekován při komunikaci pomocí modulu nRF24L01, který je stěžejní pro funkčnost celého Datového koncentrátoru. Modul byl po elektrické stránce plně funkční, avšak v rámci testování konektivity s ostatními zařízeními byl jeho dosah pro bezvýpadkové bezdrátové spojení pouze v řádech několika centimetrů, a to na všech dostupných komunikačních rychlostech (250 kb/s, 1 Mb/s, 2 Mb/s). To však neodpovídalo předpokládaným údajům diskutovaným v rámci kapitoly 2.1.1, kdy již v rámci nejvyšší možné bezdrátové komunikační rychlosti 2 Mb/s je očekávaná maximální vzdálenost dvou komunikujících zařízení v řádech jednotek metrů.

Po konzultaci tohoto problému s Ing. Baštánem a vedoucím této práce, jsem přepájel modul nRF24L01 na desce plošného spoje z polohy horizontální (viz obr. 2.17) do polohy vertikální. Otočením modulu o 90° do polohy kolmé vůči desce jsem problém vyřešil. Ten byl tedy nejspíše zapříčiněn mým nekorektním návrhem místa pro modul nRF24L01 v rámci desky plošných spojů, kdy je pod anténou původně

navrženého umístění modulu přítomno zemnění. To zjevně zapříčinilo výrazné ovlivnění parametrů antény modulu, změnu její impedance a pohlcování vyzařovaného výkonu antény.

Pro takhle opravené prototypy jsem taktéž naměřil horizontální kontrolní vyzařovací charakteristiku použitého modulu nRF24L01 pro nejvyšší komunikační rychlost 2 Mb/s a maximálním výkonem antény, která je v rámci zařízení dále využívána. Na kontrolní vyzařovací charakteristiku lze nahlédnout na obrázku 5.1, kdy ve středu je přítomen přijímač a kolem něj pod různými úhly rotuje vysílač. Ekvivalentní charakteristika byla však naměřena i v opačném případě, kdy je ve středu měření vysílač a na okraji rotuje přijímač.

Pro komunikační rychlost 2 Mb/s a nastavené maximální výkony antén je maximální vzdálenost dvou zařízení pro jejich bezvýpadkové spojení v rozmezí čtrnácti až šestnácti metrů v závislosti na orientaci obou zařízení.



Obr. 5.1: Horizontální vyzařovací charakteristika nRF24L01 (2 Mb/s)

Na základě zjištěných závad a pro vylepšení samotného zařízení by budoucí verze elektrického návrhu dle mého názoru měla obsahovat následující.

Zařízení by mělo být doplněno o speciální obvod elektrické pojistky, kterou lze nastavit ochranu před případným zkratem a nadproudem a dále pro případnou ochranu proti přepólování akumulátoru. Dále vhodným dodatkem je doplnění obvodu GPS antény o možnost připojení mimo pasivní i aktivní GPS anténu a to pomocí dopracování obvodu pro napájecí fantomové napětí, dle verze 1.3 katalogového listu modulu SIM808 [11]. Posledním vylepšením by mohlo být vynechání kontrolního napěťového děliče připojeného přímo k napětí akumulátoru. Informaci o napětí na akumulátoru je totiž možné zjistit taktéž z modulu SIM808, který je připojen k akumulátoru přímo. Takto nově uvolněný pin na mikrokontroléru ESP-WROOM-32 by mohl sloužit tvrdému vypínání modulu SIM808 u něhož je v současné době využíván pouze úsporný režim.

Z hlediska návrhu desky je nutné upravit zemnění kolem antény modulu nRF24L01 a dále upravit umístění zásuvného slotu pro SD kartu.

Dále jsem pro takto zprovozněné zařízení postupně vytvořil kompletní programového vybavení, které zastřešuje kompletní funkčnosti zařízení a je popsáno v kapitole 3. Všechny tomu příslušné soubory jsou obsaženy v příloze Diplomové práce na přiloženém CD.

Mimo vytvoření hlavního programu a s tím souvisejících knihoven jsem taktéž navrhl vlastní nadstavbový lightweight protokol pro komunikaci mezi Datovým koncentrátorem a snímači (viz kapitola 3.3.4). Dále jsem navrhl formát měřicího XML souboru pro strukturované ukládání naměřených dat (viz kapitola 3.3.3).

V neposlední řadě jsem v rámci mnou vybrané cloudové platformy ThingSpeak vytvořil měřicí kanály pro Data koncentrátor (viz kapitola 4.4) a zprovoznil jsem komunikaci mezi datovým koncentrátorem a cloudovou platformou (viz kapitola 4.3). Taktéž jsem v rámci platformy pro účely zařízení vytvořil vlastní softwarovou komponentu pro sledování zařízení v reálném čase (viz kapitola 4.4.3) a pro ukázku vytvořil a použil výpočetní skript z prostředí Matlab pro zpracování naměřených dat (viz kapitola 4.4.2), na který lze nahlédnout v příloze H.

S takto kompletovaným zařízením, jsem provedl několik zkušebních měření ve všech jeho různých módech, které toto zařízená nabízí. Tedy v módu kompletním zařízení nabízí komunikaci se snímači, dále ukládání dat na SD kartu, extrakci dat na cloud a taktéž měření aktuální polohy pomocí GPS. V dalších módech jsou některé tyto funkcionality nevyužity.

Výsledky jedno takovéto provedeného zkušebního měření jsou vyobrazeny v rámci kapitoly 4. V rámci obrázku 4.4 jsou vyobrazena naměřená data z připojených snímačů, dále na obrázku 4.5 jsou vidět číselné hodnoty vypočteného aritmetického průměru posledních čtyřech naměřených hodnot na základě ve ThingSpeaku vytvo-

řeného výpočetního skriptu a na obrázcích 4.2, 4.6 a 4.7 je možné vidět vykonanou trasu v rámci tohoto zkušebního měření.

Toto měření se sestávalo ze tří prototypových zařízení Datového koncentrátoru, přičemž jedno zařízení zastávalo v rámci měření pozici plnohodnotného Datového koncentrátoru se všemi jeho funkcemi a další dvě pozici pseudosnímačů, které zasílají každých třicet sekund uměle generovaná data, v tomto případě pilovitý signál v rozmezí 0-90 a 0-190 s přírůstkem 10. Soubory programového vybavení pseudosnímačů je obsaženo na k práci přiloženém CD.

Kompletní záznam z toho měření uložený zařízením na SD kartu v podobě XML souboru je uveden v rámci přílohy G nebo na přiloženém CD.

Ze výše zmíněných obrázků a XML souboru z kontrolního měření lze vyvodit, že na cloud byly nahrány nahrány všechny v rámci měření ze snímačů zaslané hodnoty. K tomu však dojde pouze tehdy, pokud dané snímače měří s periodou větší jak 30 sekund. Toto omezení vychází z aktuálně využívané licence v rámci platformy ThingSpeak, která má v sobě omezující faktor, že nahrát data na cloud může uživatel s nejkratším rozmezí 15ti sekund. Pokud tedy taktéž nahráváme data do dvou měřicích kanálů (Kanál GPS polohy a Kanál měřicích uzlů) můžeme tak činit s nejmenší periodou 30ti sekund. Řešením tohoto problému je samozřejmě zakoupení adekvátní licence k požadovanému typu měření.

Závěr

Tato diplomová práce navazuje na mou semestrální práci [1] a zabývala se koncepcí a realizací Datového koncentrátoru, dále jeho elektrickým návrhem včetně výběru vhodných komponent a kompletním programovým vybavením pro zprovoznění celého zařízení dle jeho původní navržené koncepce. Pro zařízení byl taktéž vytvořen měřicí profil v rámci vybrané cloudové platformy ThingSpeak a jeho prostředí přizpůsobeno k nahrávání měřených hodnot z datového koncentrátoru. Pomocí zařízení bylo provedeno několik kontrolních měření, přičemž jedno je konkrétněji diskutováno v kapitole 5.

V rámci kapitoly 1 bylo pojednáno o samotné koncepci datového koncentrátoru a jeho významu, dále o jeho obecném principu, vysvětlení co je důvodem vzniku takovéhoto zařízení a proč nebylo navázáno na některé již existující systémy. Dále došlo k vydefinování hardwarových požadavků na takovéto zařízení. V neposlední řadě zde taktéž byly zmíněny softwarové nástroje, které byly použity v rámci realizace elektrického návrhu, programového vybavení či samotné fyzické realizace Datového koncentrátoru.

Následující kapitola (kapitola 2) se zabývá konkrétním výběrem vhodného hardwaru na základě vydefinovaných požadavků v rámci kapitoly 1 a následnou diskuzí ohledně jejich parametrů. Dále je v této kapitole ukázána výsledná fyzická realizace fyzického hardwaru Datového koncentrátoru, na kterou lez nahlédnout v rámci obrázcích 2.17 a obrázku 2.18 a v sekci 2.3 je vedena diskuze o návrhu elektrického schématu pro Datový koncentrátor na které lze taktéž nahlédnout v rámci přílohy A. Taktéž je v této kapitole rozebráno samotné rozvržení fyzických komponent na desce plošných spojů (viz příloha B).

Kapitola 3 pojednává o vytvoření kompletního programového vybavení pro výpočetní jádro Datového koncentrátoru. Je zde zmíněno, že Datový koncentrátor je založen na operačním systému reálného času FreeRTOS a v rámci kapitoly 3.3 je podrobně vysvětlená hierarchie programu, který právě možnosti FreeRTOS využívá a dále v něm obsažené jednotlivé úlohy. V této kapitole je taktéž podrobně rozebrán mnou navržený lightweight komunikační protokol, definující komunikaci i v ní obsažené rámce pro spojení se snímači. Dále je zde taktéž podrobně rozebrán mnou navržený formát souboru XML pro strukturované ukládání dat z měření na lokální úložiště v podobě SD karty. Dále je v kapitole 3.4 vysvětlena hierarchie a návaznost jednotlivých tříd a tedy datových objektů navržených pro jednoduchou správu dat a informací v rámci měření. Tuto skutečnost popisující zjednodušený UML diagram je vyobrazen na obrázku 3.5. Poslední část kapitoly 3 (viz kapitola 3.5) je věnována objasnění struktury použitých knihoven, jejich tvorbě a principiálnímu popisu.

V rámci předposlední kapitoly 4 je pojednáno o vybrané cloudové platformě pro

Datový koncentrátor a o důvodech, na základě kterých byla tato platforma upřednostněna před ostatními. V rámci kapitoly je taktéž podrobněji rozebráno prostředí této platformy, jeho možnosti, použití pro měření a rozhraní v rámci kterého s prostředím komunikuje samotný Datový koncentrátor.

V poslední kapitole textu 5 je rozebrán celý postup, který byl absolvován pro vytvoření konsolidovaného zařízení Datového koncentrátoru. Dále jsou v kapitole zevrubně diskutovány problémy, kterým jsem čelil v rámci realizace Datového koncentrátoru a jejich řešení a případné návrhy pro budoucí realizaci. V rámci kapitoly je taktéž diskutována horizontální vyzařovací charakteristika zařízení pro komunikaci pomocí modulu nRF24L01 (viz obr. 5.1). V neposlední řadě je zde podrobněji zmíněno testovací měření, které ověřuje funkčnost realizovaného zařízení ve všech požadovaných směrech. Na XML soubor z toho měření lze nahlédnout v příloze G, na nahraná data do cloudového prostředí na obrázcích v rámci kapitoly 4, případně na kompletní stránky z měřicích kanálů v rámci přílohy I.

Výsledkem této diplomové práce jsou tedy tři odzkoušené a zprovozněné prototypy Datového koncentrátoru. Nejdříve byla rozmyšlena koncepce samotného zařízení spolu s vydefinováním požadavků na toto zařízení. Na základě tohoto byly vybrány vhodné komponenty, bylo vytvořené elektrické schéma, následně byl vytvořen návrh desky plošných spojů, který se zadal do výroby. Tyto vyrobené desky plošných spojů byly ručně osazeny vybranými komponentami. Takováto kompletní fyzická realizace byla následně odzkoušena základním programovým vybavením nahraným do MCU modulu a jednoduchým měřením základních elektrických veličin. Dále bylo pro zařízení vytvořeno kompletní programové vybavení založené na FreeRTOS, které umožňuje na základě konfiguračního souboru vloženém na lokálním úložišti nastavit základní parametry měření a připojit k Datovému koncentrátoru konfigurované senzorické jednotky pomocí rozhraní nRF24L01 s nadstavbovým lightweight komunikačním protokolem a koncentrovat od nich naměřená data. Dále tato data ukládat v přesně definovaném formátu do hlavního měřicího XML souboru. Pro koncentrovaná data bylo taktéž bylo v rámci platformy ThignSpeak vytvořeno prostředí, kde je možné tyto data ukládat, analyzovat a vizualizovat v reálném čase.

Veškerá dokumentace k Datovému koncentrátoru je obsažena v příloze této diplomové práce a na přiloženém CD, jež obsahuje kompletní elektrické schéma Datového koncentrátoru, dále návrh DPS Datového koncentrátoru, programové vybavení pro kompletní odzkoušení hardwaru Datového koncentrátoru, dále programové vybavení pro celkovou funkčnost Datového koncentrátoru včetně všech potřebných knihoven a programové vybavení pro vytvoření pseudosnímačů pro testovací měření. Dále jsou v příloze přítomny skript pro analýzu dat v prostředí ThingSpeak a všechny potřebné soubory obsahující zdrojový kód pro vytvoření vlastní softwarové komponenty v rámci ThingSpeaku pro sledování zařízení v reálném čase.

Budoucí vize zařízení spočívá v dopracování preferovaného připojení k internetu pomocí lokálních sítí, tedy zařízení bude doplněno o funkce pro připojení pomocí Wi-Fi a toto připojení bude preferováno, před připojením pomocí protokolu GPRS v rámci modulu SIM808. Důvodem je redukce finanční náročnosti zařízení pro spojení s cloudem a zvýšení rychlosti připojení.

Dalším možným milníkem je vytvoření mobilní aplikace Datového koncentrátoru pro správu konfigurace zařízení a aktuálního měření. Datový koncentrátor by byl s mobilním zařízením spojen pomocí rozhraní Bluetooth. Dále by pak platforma mohla sloužit i jako prostředník pro nahrávání naměřených dat do cloudového prostředí ThingSpeak.

Již realizovaná práce s budoucí vizí zařízení je zasazena do kontextu v rámci vizualizace na obrázku 1.1.

Literatura

[1] *DVORSKÝ, Petr. Datový koncentrátor.* Brno, 2020, 55 s. Semestrální práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce: doc. Ing. Petr Fieldler, Ph.D.

[2] *ZM6380 Data Concentrator Unit: A DCU-CUM-MODBUS CONVERTER FOR EFFICIENT & COST-SAVING 2-WAY DATA COMMUNICATION* [online]. Zodicom, 2018 [cit. 2021-04-22]. Dostupné z URL: <http://zodicom.com/Brochure/ZM6380_OZ.pdf>.

[3] *MEHAFFEY, Joe, Jack YEAZEL, Sam PENROD a Allory DEISS. Error Measures. Gpsinformation.net* [online]. [cit. 2020-12-25]. Dostupné z URL: <http://gpsinformation.net/main/errors.htm>.

[4] *Český telekomunikační úřad: Využívání vymezených rádiových kmitočtů* [online]. 2018 [cit. 2020-12-25]. Dostupné z URL: <https://www.ctu.cz/vyuzivani-vymezenych-radiovych-kmitoctu>.

[5] *Autodesk: EAGLE* [online]. Autodesk, 2020 [cit. 2020-12-25]. Dostupné z URL: <https://www.autodesk.com/products/eagle/overview>.

[6] *WEBENCH® POWER DESIGNER: Texas Instruments* [online]. 2020 [cit. 2020-12-25]. Dostupné z URL: <https://webench.ti.com/power-designer/>.

[7] *Platform IO* [online]. PlatformIO Labs OÜ [cit. 2020-12-25]. Dostupné z URL: <https://platformio.org/>.

[8] *JLCPCB: PROTOTYPE AND SMALL SERIES Reliable PCB Services* [online]. JLCPCB.COM, 2020 [cit. 2020-12-25]. Dostupné z URL: <https://jlcpcb.com/>.

[9] *MACTRONICA: Electrónica y Technología* [online]. [cit. 2020-12-25]. Dostupné z URL: <https://www.mactronica.com.co/modulo-transceptor-nrf24l01-smd-mini>.

[10] *NRF24L01+ - Single chip 2.4Ghz Transceiver: Preliminary product specification v1.0* [online]. NORDIC Semiconductors [cit. 2020-12-25]. Dostupné z URL: <https://www.sparkfun.com/datasheets/Components/SMD/nRF24L01Pluss_Preliminary_Product_Specification_v1_0.pdf>.

[11] *SIM808 Hardware Design V1.03* [online]. Shanghai SIMCom Wireless Solutions, 2016 [cit. 2020-12-25]. Dostupné z URL: <https://simcom.ee/documents/SIM808/SIM808_Hardware%20Design_V1.03.pdf>.

[12] *GNSS (GPS) ACCURACY EXPLAINED. Jinuper Systems* [online]. Juniper System, 2020 [cit. 2020-12-25]. Dostupné z URL: <https://junipersys.com/index.php/support/article/6614>.

[13] *SD SPI Host Driver: ESP-IDF Programming Guide. Espressif.com* [online]. Espressif Systems (Shanghai) CO., 2020 [cit. 2020-12-25]. Dostupné z URL: <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/peripherals/sdspi_host.html>.

[14] *SDMMC Host Driver: ESP-IDF Programming Guide. Espressif.com* [online]. Espressif Systems (Shanghai) CO., 2020 [cit. 2020-12-25]. Dostupné z URL: <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/peripherals/sdmmc_host.html>.

[15] *Description of the FAT32 File System. Support.microsoft.com* [online]. Microsoft, 2018 [cit. 2020-12-25]. Dostupné z URL: <https://support.microsoft.com/en-us/help/154997/description-of-the-fat32-file-system>.

[16] *SD/SDHC/SDXC Specifications and Compatibility: SanDisk® Global Customer Support* [online]. Western Digital Corporation [cit. 2021-5-4]. Dostupné z URL: <https://kb.sandisk.com/app/answers/detail/a_id/2520/~/sd%2Fsdhc%2Fsdxc-specifications-and-compatibility#:~:text=There%20are%20three%20main%20types,are%20different%20with%20each%20type.>.

[17] *DS3231: Extremely Accurate I2C-Integrated RTC/TCXO/Crystal* [online]. Maxim Integrated, 2015 [cit. 2020-12-25]. Dostupné z URL: <https://datasheets.maximintegrated.com/en/ds/DS3231.pdf>.

[18] *ESP32-WROOM-32: Datasheet Version 3.0. Espressif.com* [online]. Espressif Systems (Shanghai) Co. [cit. 2020-12-25]. Dostupné z URL: <https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf>.

[19] *ESP32 Series: Datasheet Version 3.4. Espressif.com* [online]. Espressif Systems (Shanghai) Co. [cit. 2020-12-25]. Dostupné z URL:

<https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf>.

[20] *DevKits Technical Documentation: ESPRESSIF. Espressif.com* [online]. Espressif Systems (Shanghai) CO., 2021 [cit. 2021-4-18]. Dostupné z URL: <https://www.espressif.com/en/support/download/documents/development-board?keys=&field_type_tid%5B%5D=127>.

[21] *SD Pull-up Requirements: ESP-IDF Programming Guide. Espressif.com* [online]. Espressif Systems (Shanghai) CO., 2020 [cit. 2021-4-28]. Dostupné z URL: <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/peripherals/sd_pullup_requirements.html#solutions>.

[22] *Getting Started With ESP32. Electronics-lab.com: Open Source hardware projects* [online]. 2019 [cit. 2020-12-26]. Dostupné z URL: <https://www.electronics-lab.com/getting-started-esp32/>.

[23] *FT232R USB UART IC Datasheet: Version 2.16. Ftdichip.com* [online]. Future Technology Devices International, 2020 [cit. 2020-12-25]. Dostupné z URL: <https://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT232R.pdf>.

[24] *INR18650-35E SAMSUNG SDI: Speification of Product - version No. 1.0. Tme.eu* [online]. SAMSUNG SDI Co., 2016 [cit. 2020-12-25]. Dostupné z URL: <https://www.tme.eu/Document/f9d93f9cb60fa6eb0a4caf9df5c96711/ACCU-INR18650-35E.pdf>.

[25] *Samsung INR18650-35E 3500mAh (Pink) Review. Lygte-info.dk* [online]. 2016 [cit. 2020-12-25]. Dostupné z URL: <https://lygte-info.dk/review/batteries2012/Samsung%20INR18650-35E%203500mAh%20%28Pink%29%20UK.html>.

[26] *KAZDA, Tomáš. Vliv nízké teploty na kapacitu vybraných Li-ion akumulátorů. In: TZB-info* [online]. Fakulta elektrotechniky a komunikačních technologií VUT v Brně, Ústav elektrotechnologie, 2021, 4.1.2021 [cit. 2021-04-22]. Dostupné z URL: <https://oze.tzb-info.cz/akumulace-elektriny/21683-vliv-nizke-teploty-na-kapacitu-vybranych-li-ion-akumulatoru>.

[27] *TLV62569 2-A High Efficiency Synchronous Buck Converter in SOT Package. Ti.com* [online]. Texas Instruments, 2017 [cit. 2020-12-25]. Dostupné z URL:

<https://www.ti.com/lit/ds/symlink/tlv62569.pdf?ts=1608471629944&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FTLV62569>.

[28] *MCP73831/2: Miniature Single-Cell, Fully Integrated Li-Ion, Li-Polymer Charge Management Controllers* [online]. Microchip Technology, 2014 [cit. 2020-12-26]. Dostupné z URL: <https://cdn.sparkfun.com/assets/learn_tutorials/6/9/5/MCP738312.pdf>.

[29] *I2C: I2C at the Hardware Level* [online]. SparkFun Electronics ®, 2013 [cit. 2021-4-29]. Dostupné z URL: <https://learn.sparkfun.com/tutorials/i2c?_ga=2.253260361.1339946030.1619710424-535167152.1613252231>.

[30] *Adafruit FONA 808 Cellular + GPS Breakout: A full-featured GSM/GPRS module with GPS as well!* [online]. 2015: Adafruit [cit. 2021-4-29]. Dostupné z URL: <https://learn.adafruit.com/adafruit-fona-808-cellular-plus-gps-breakout/downloads>.

[31] *Build System: ESP-IDF Programming Guide. Espressif.com* [online]. Espressif Systems (Shanghai) CO., 2020 [cit. 2021-04-10]. Dostupné z URL: <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/build-system.html>.

[32] *Use Arduino as a component of ESP-IDF: esp32-arduino-lib-builder* [online]. 2020 [cit. 2021-04-10]. Dostupné z URL: <https://github.com/espressif/arduino-esp32/blob/master/docs/esp-idf_component.md>.

[33] *BARRY, Richard. Mastering the FreeRTOS™ Real Time Kernel: A Hands-On Tutorial Guide. Freertos.org* [online]. Real Time Engineers, 2016 [cit. 2021-04-18]. Dostupné z URL: <https://www.freertos.org/fr-content-src/uploads/2018/07/161204_Mastering_the_FreeRTOS_Real_Time_Kernel-A_Hands-On_Tutorial_Guide.pdf>.

[34] *ESP-IDF FreeRTOS SMP Changes: ESP-IDF Programming Guide. Espressif.com* [online]. Espressif Systems (Shanghai) CO., 2021 [cit. 2021-04-18]. Dostupné z URL:

<https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/freertos-smp.html>.

[35] *Introduction to RTOS - Solution to Part 12 (Multicore Systems)* [online]. Maker.io, 2020 [cit. 2021-04-18]. Dostupné z URL: <https://www.digikey.be/en/maker/projects/introduction-to-rtos-solution-to-part-12-multicore-systems/369936f5671d4207a2c954c0637e7d50>.

[36] *Learn More About ThingSpeak: ThingSpeak™* [online]. The MathWorks, 2021 [cit. 2021-04-19]. Dostupné z URL: <https://thingspeak.com/pages/learn_more>.

[37] *ThingSpeak Documentation: The IoT Platform with MATLAB Analytics* [online]. The MathWorks, 2021 [cit. 2021-04-19]. Dostupné z URL: <https://www.mathworks.com/help/thingspeak/>.

[38] *API Reference: Use the REST and MQTT APIs to update ThingSpeak™ channels and to chart numeric data stored in channels* [online]. The MathWorks, 2021 [cit. 2021-04-19]. Dostupné z URL: <https://www.mathworks.com/help/thingspeak/channels-and-charts-api.html>.

[39] *REST API: Use REST API calls to create and update ThingSpeak™ channels and charts* [online]. The MathWorks, 2021 [cit. 2021-04-19]. Dostupné z URL: <https://www.mathworks.com/help/thingspeak/rest-api.html>.

Seznam symbolů a zkratek

Symboly:

Δ	Absolutní chyba měření [-]
δ	Relativní chyba měření [-, ppm, %]
T, t, τ	Čas [s]
U	Elektrické napětí [V]
I	Elektrický proud [A]
R	Elektrický odpor [Ω]
Q	Elektrický náboj [C, As]
C	Elektrická kapacita [F]

Veličiny:

V	Volt
A	Ampér
s	Vteřina
Ω	Ohm
Ah	Ampérhodina
C	Coulomb
F	Farad
m	Metr
m/s	Metr za sekundu
km/h	Kilometr za hodinu
Hz	Hertz
°C	Stupeň Celsia
°	Úhlový stupeň
B	Byte

b Bit

b/s Bit Per Second - Bitů za sekundu

ppm Parts Per Milion - Dílů na jeden milión

Zkratky:

CSV Souborový formát, který hodnoty odděluje čárkami – Comma-Separated Values

XML Rozšířitelný značkovací jazyk – Extensible Markup Language

JSON JavaSkriptový objektový zápis – JavaScript Object Notation

XLSX Nástupce formátu XLS – Microsoft Excel Open XML Spreadsheet Format

XLS Přípona souborů specifikace Office Open XML pro Microsfot Excel – Old Microsoft Excel Open XML Spreadsheet Format

CEP (50 %) Pravděpodobnost kruhové chyby udávající pravděpodobnost 50ti% výskytu daného tělesa v rámci udaného poloměru s počátkem na aktuálně změřeném souřadnicovém bodu – Circural Error Probable

ISM Pro účely vědy, techniky a zdravotnictví - Industrial, Scientific And Medical Purposes

GPRS Paketově orientovaný mobilní datový protokol v rámci mobilní sítě (GSM) – General Packet Radio Service

GSM Globální systém pro mobilní komunikaci – The Global System For Mobile Communications

GPS Globální polohový systém – Global Positioning System

GNSS Globální družicový polohový systém – Global Navigation Satellite System

IoT Internet věcí – Internet Of Things

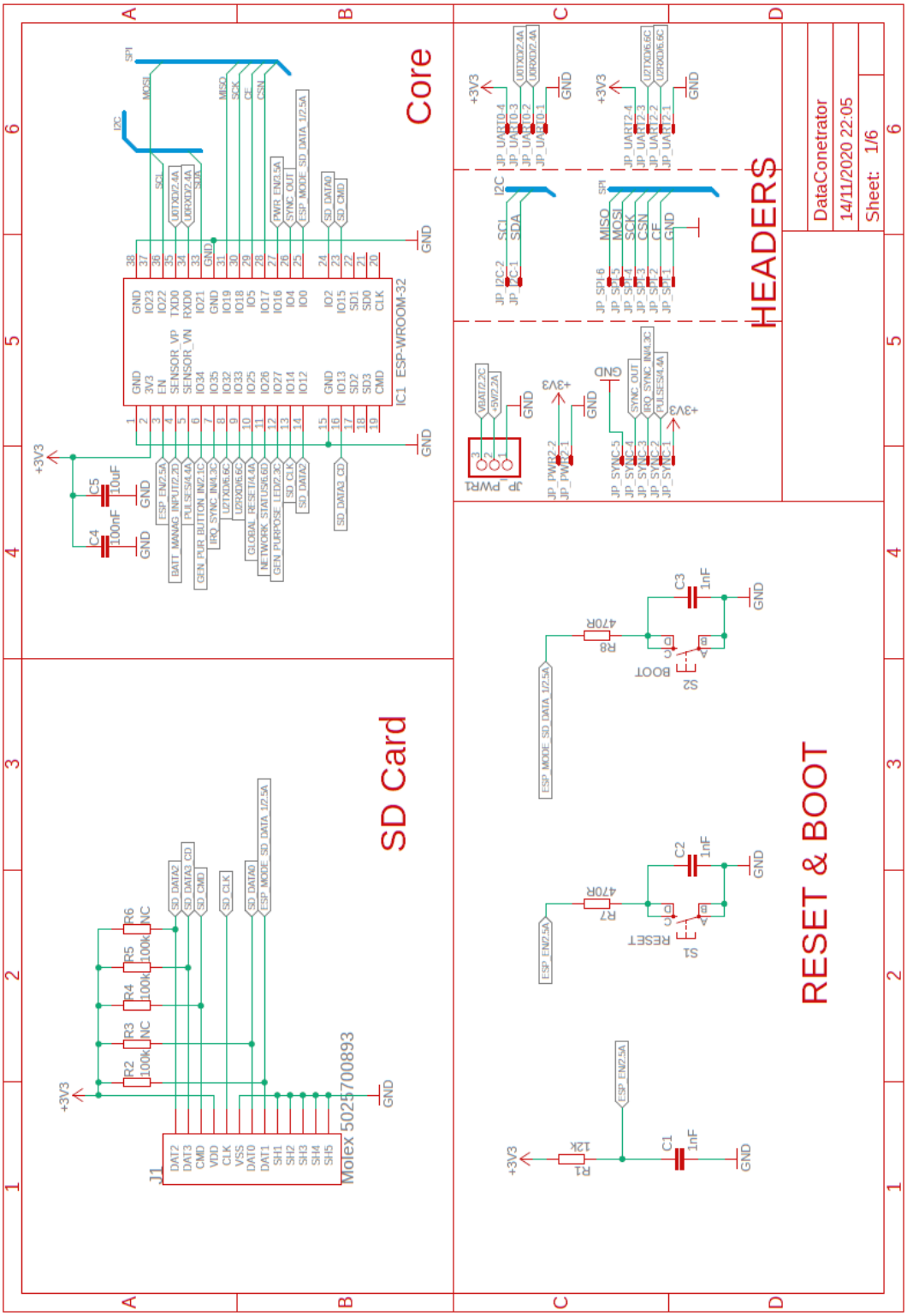
HTTP Internetový protokol pro komunikaci s WWW servery – Hypertext Transfer Protocol

MQTT Komunikační protokol pro odlehčený přenos dat často používaný v IoT – Message Queuing Telemetry Transport

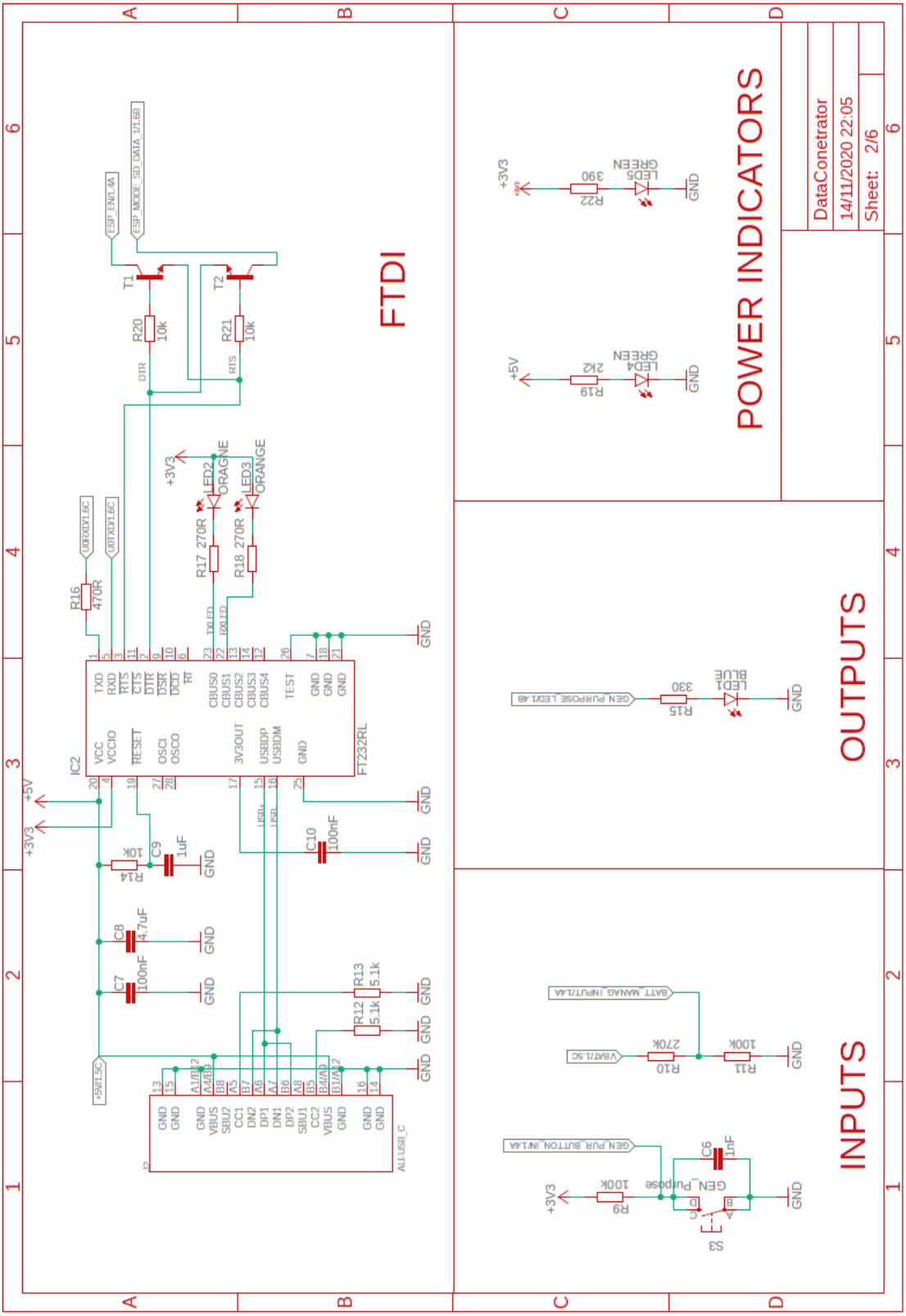
EDA	Softwarový nástroj pro návrh elektronických zařízení – Electronic Design Automation
DPS	Deska plošných spojů
PCB	Deska plošných spojů – Printed Circuit Board
IDE	Integrované vývojové prostředí – Integrated Developmend Enviroment
VS code	Softwarový nástroj Microsoft Visual Studio Code
SMD	Součástka určena pro povrchovou montáž – Surface Mount Device
MCU	Mikrokontrolér – Microcontroller unit
SRAM	Statická pamět s náhdoným přístupem – Static Random Access Memory
ROM	Paměť pouze pro čtení – Read Only Memory
PWM	Pulzně šířková modulace – Pulse Width Modulation
BT	Bluetooth
BLE	Bluetooth low energy
UART	Asynchronní sériové rozhraní pro přenos dat mezi zařízeními v obou směrech – Universal Asynchronous Receiver/Transmitter
I2C	Sériové komunikační rozhraní – Inter-Integrated Circuit
SPI	Sériové periferní rozhraní – Serial Peripheral Interface
USB	Univerzální sériová sběrnice – Universal Serial Bus
MOSI	Master Out Slave In
MISO	Master In Slave Out
CRC	Cyklický redundantní součet – Cyclic Redundancy Check
ESD	Elektrostatický výboj – Electrostatic Discharge
SMP	Symetrický multiprocesing – Symmetric Multiprocessing
NTP	Protokol pro synchronizaci vnitřních hodin počítačů – Network Time Protocol

Seznam příloh

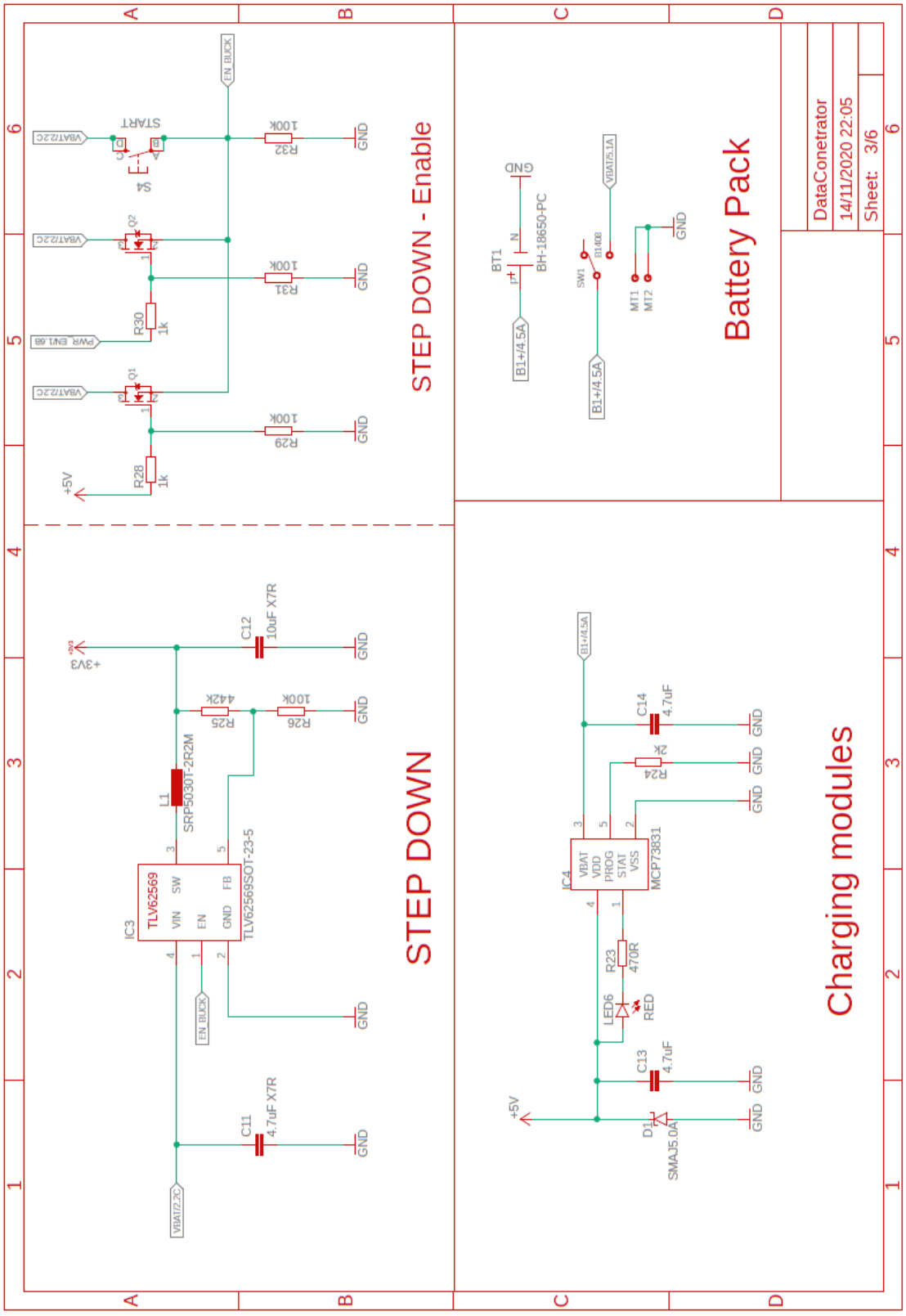
A El. schéma Datového koncentrátoru



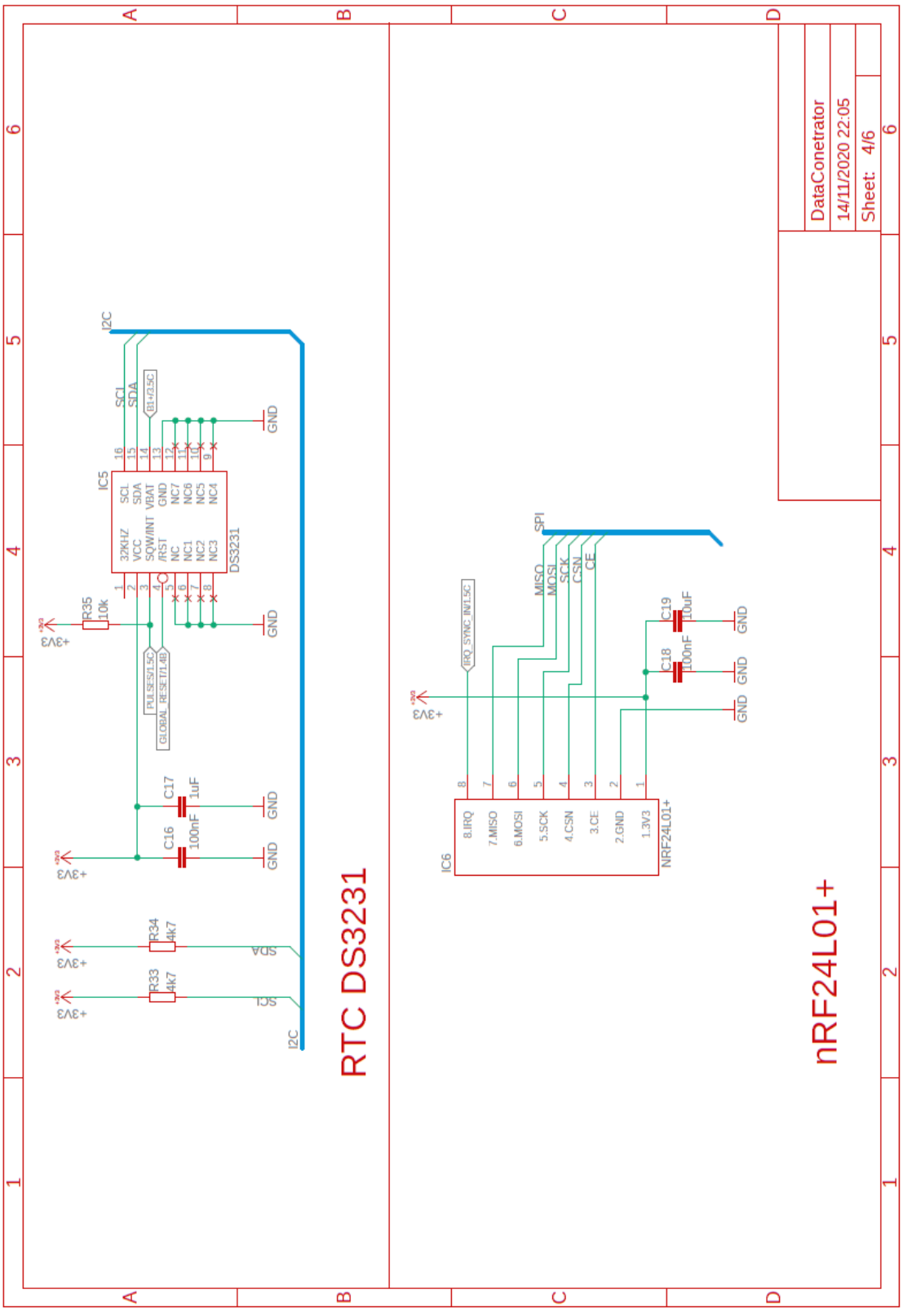
Obr. A.1: Elektrotechnické schéma Datového koncentrátoru - list č.1



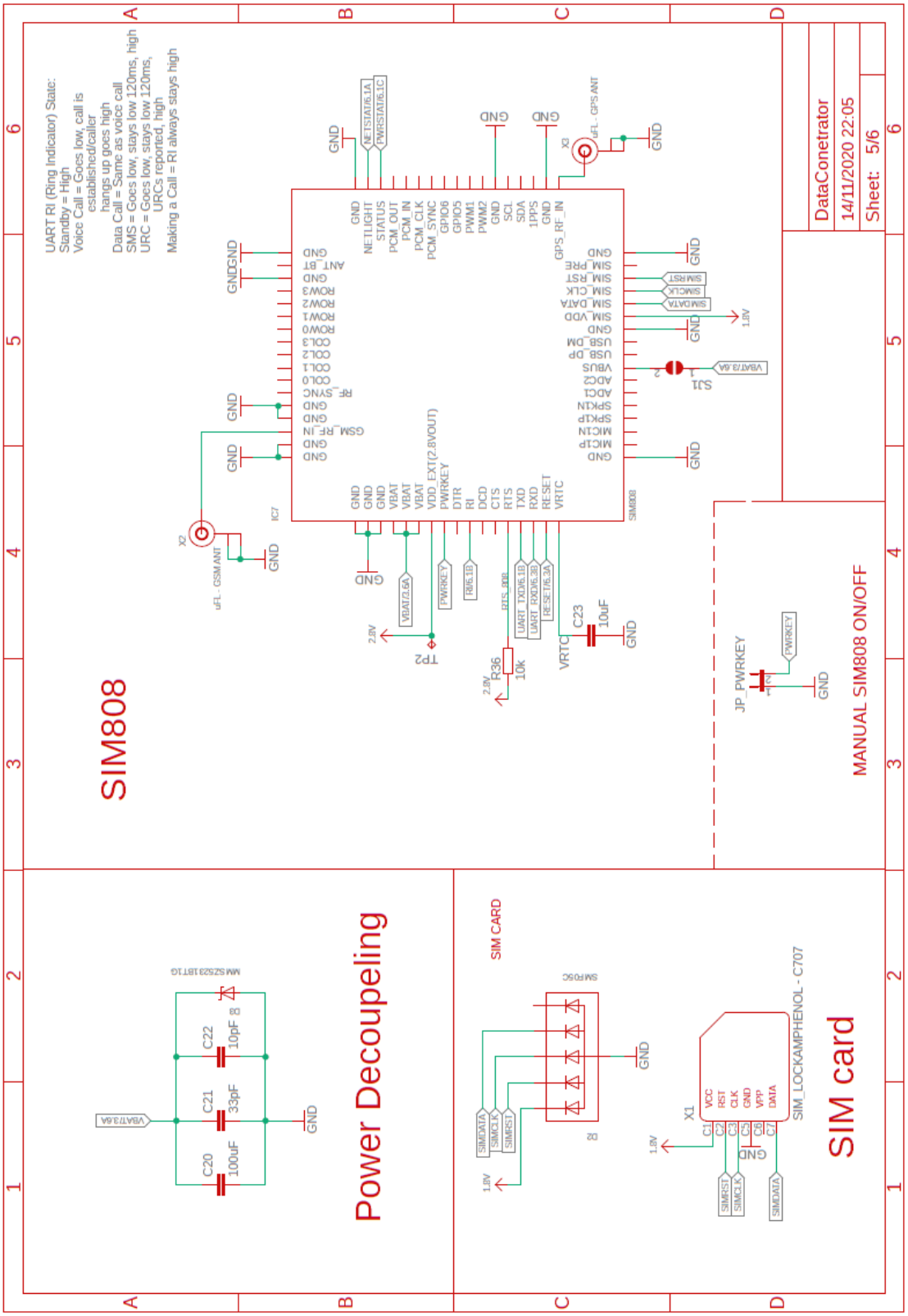
Obr. A.2: Elektrotechnické schéma Datového koncentrátoru - list č.2



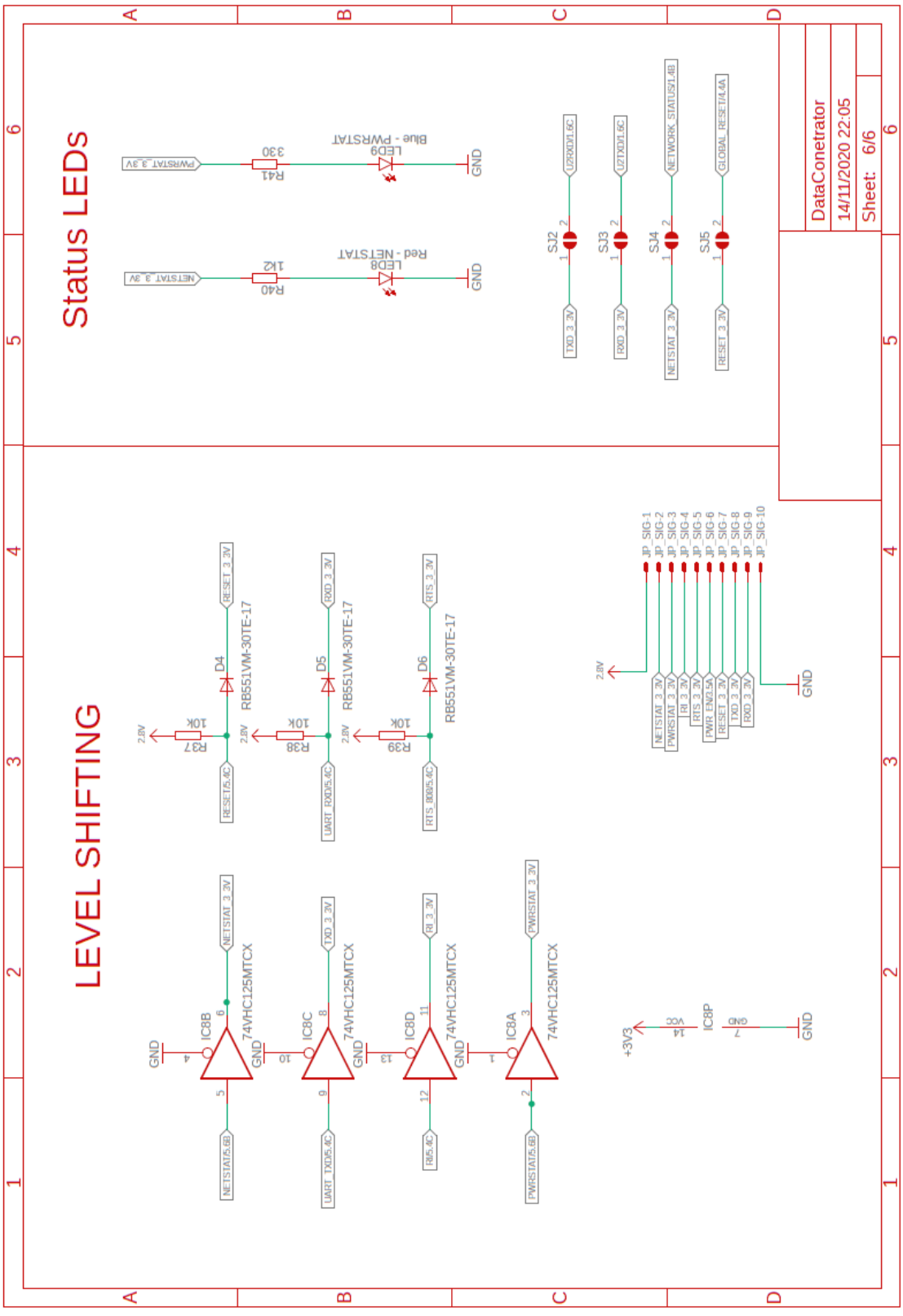
Obr. A.3: Elektrotechnické schéma Datového koncentrátoru - list č.3



Obr. A.4: Elektrotechnické schéma Datového koncentrátoru - list č.4

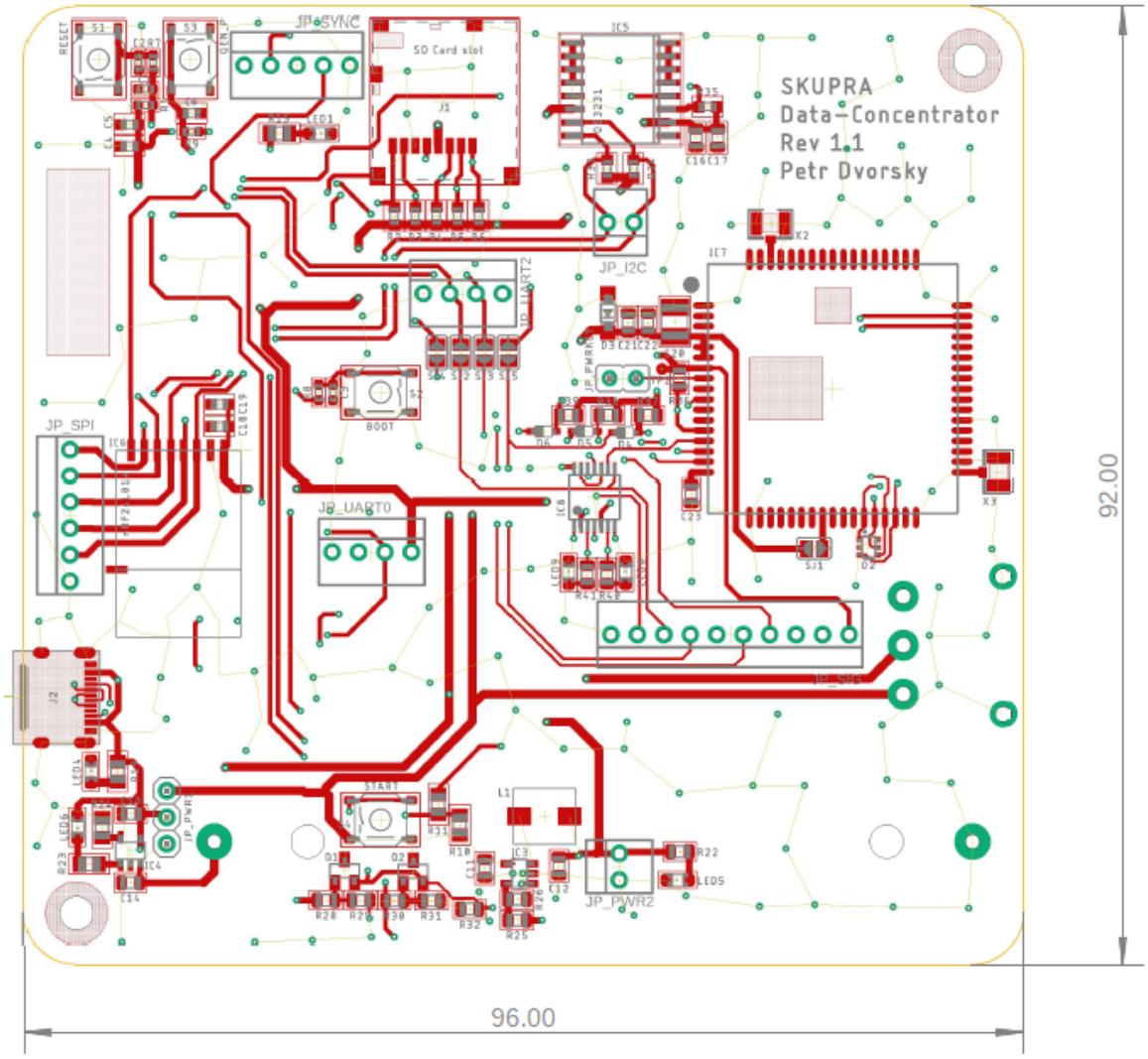


Obr. A.5: Elektrotechnické schéma Datového koncentrátoru - list č.5

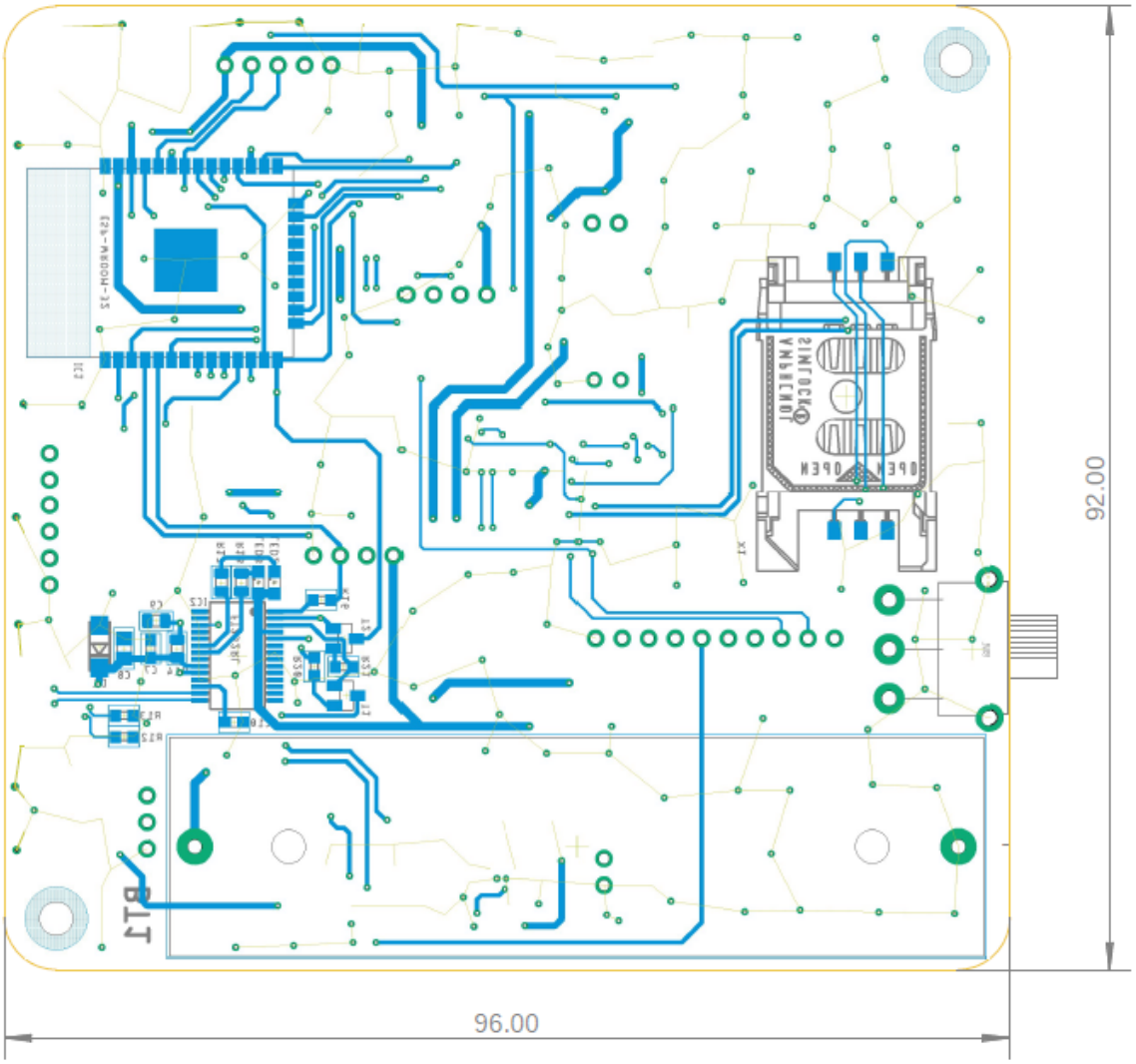


Obr. A.6: Elektrotechnické schéma Datového koncentrátoru - list č.6

B Deska plošných spojů Datového koncentrátoru

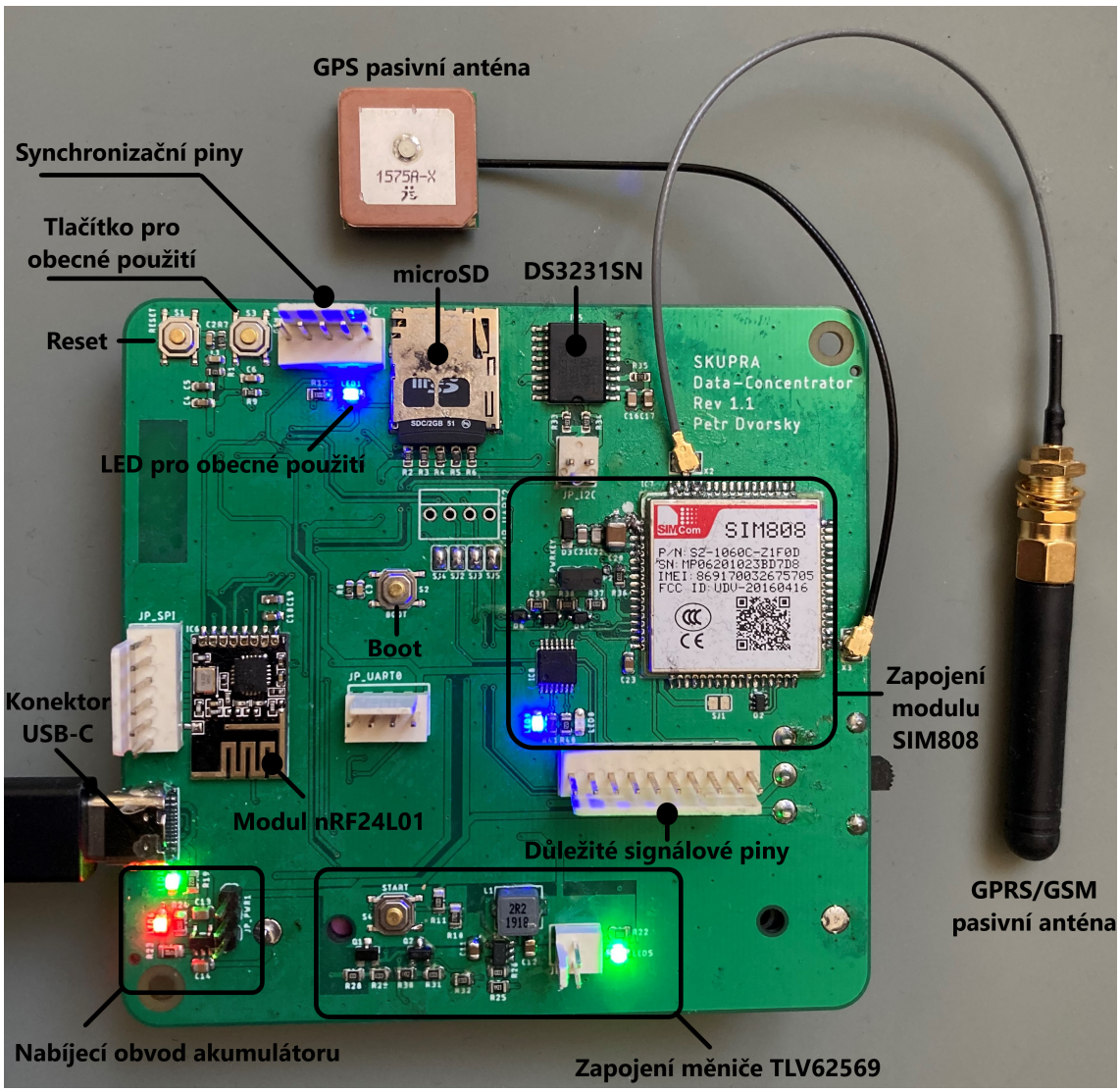


Obr. B.1: Přední strana DPS Datového koncentrátoru - Verze 1.1

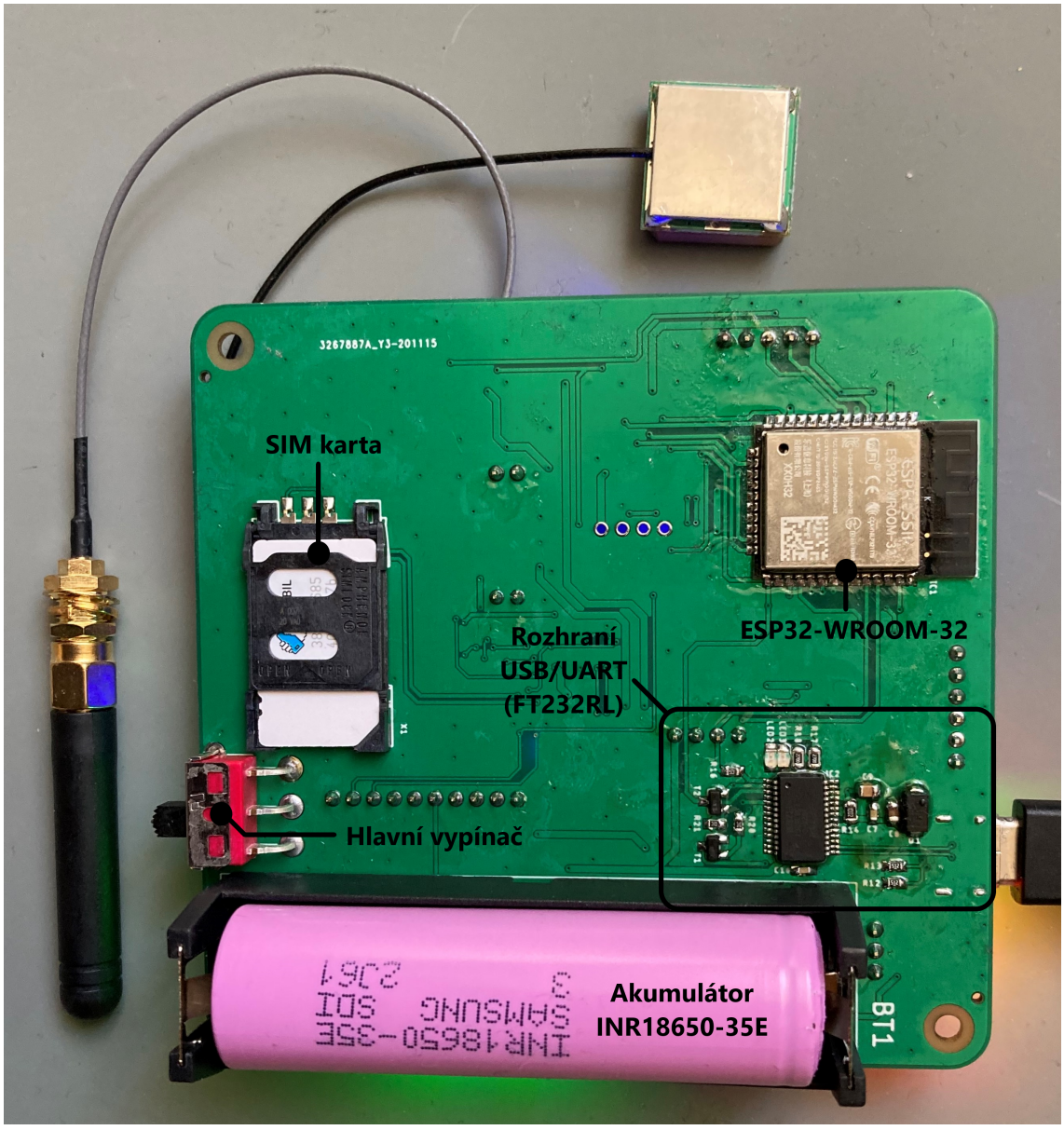


Obr. B.2: Zadní strana DPS Datového koncentrátoru - Verze 1.1

C Fyzická podoba Datového koncentrátoru

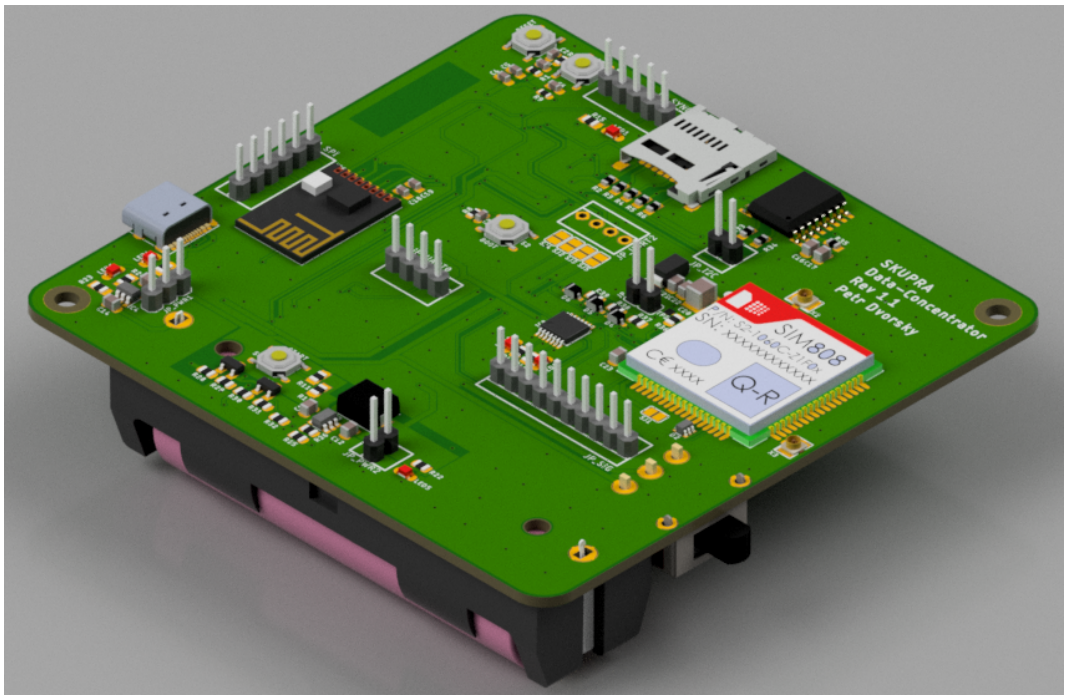


Obr. C.1: Osazená DPS Datového koncentrátoru - Přední strana

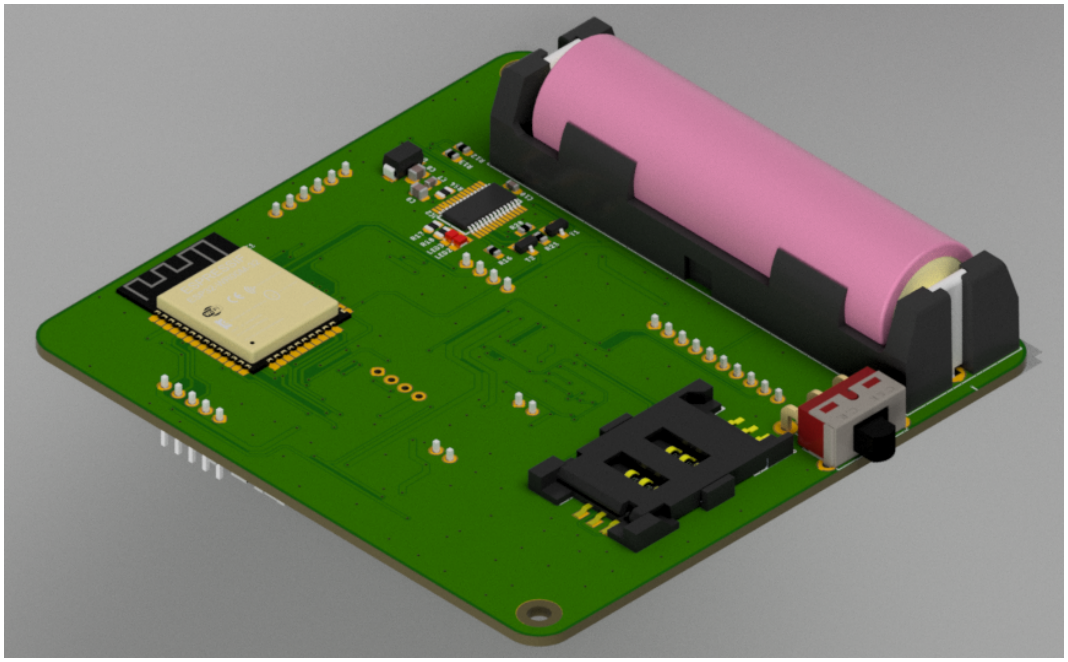


Obr. C.2: Osazená DPS Datového koncentrátoru - Zadní strana

D 3D model Datového koncentrátoru



Obr. D.1: 3D model DPS Datového koncentrátoru - Přední strana



Obr. D.2: 3D model DPS Datového koncentrátoru - Zadní strana

E Lightweight protokol - Význam polí v komunikačních rámcích

Tab. E.1: Přehled významů polí v kontrolním rámci

Pole	Byty	Hodnota	Popis
Délka	1	0-32	Počet bytů obsažených v rámci
ID	1	0-255	ID aktuálního rámce
CMD	1	'C'	Označení kontrolního rámce
Sériové číslo	1	0-255	Sériové číslo snímače
Verze protokolu	1	0-255	Verze aktuálního protokolu
Deskriptor	1	'?'	Detekce připojení
		'T'	Start měření
		'S'	Stop měření
		'K'	OK potvrzení
Data	0-26	-	Vyplnění zaslaného času viz 3.3

Tab. E.2: Přehled významů polí v datového rámci

Pole	Byty	Hodnota	Popis
Délka	1	0-32	Počet bytů obsažených v rámci
ID	1	0-255	ID aktuálního rámce
CMD	1	'D'	Označení datového rámce
Sériové číslo	1	0-255	Sériové číslo snímače
Verze protokolu	1	0-255	Verze aktuálního protokolu
Datový typ	1	0	Prázdný
		1	Bool
		2	Char
		3	Short
		4	Int
		5	Float
Multiplikátor	1	0	Prázdný
		1	Nano
		2	Mikro
		3	Mili
		4	Centi
		5	Deci
		6	Deka
		7	Hekto
		8	Kilo
		9	Mega
		10	Giga
		11	Tera
Fyzikální jednotka	1	0	Prázdný
		1	[s] - Sekunda
		2	[m] - Metr
		3	[kg] - Kilogram
		4	[A] - Ampér
		5	[°C] - Stupeň Celsia
		6	[V] - Volt
		7	[Pa] - Pascal
		8	[N] - Newton
		9	[m/s] - Metr za sekundu
Měřená hodnota	4	-	Naměřená hodnota viz obr. 3.4
Časová značka	8	-	Časová značka viz obr. 3.4

F Příklad konfiguračního souboru pro nastavení měření

```
Config
Warning: "Please follow exact data format of config file. In other cases Data Concentrator (DC) will not work properly."
Note: "If you not sure about your settings follow this TEST FILE"
"Dollar" = Modules TurnON-(1)/TURN OFF-(0)
         = nRF24L01/GPS/CloudCommunication/SD Card logging
         - example = 1/0/1/0 - nRF24L01=ON, GPS = OFF, CloudCommunication = ON, SD CARD logging = OFF
"Hashtag" = Number of Nodes
"Arrow" = Nodes settings format
Format = Data_Type/Value_Type/Unit/Device_Serial/Device_Type/Slave_Address/Master_Address/Device_Description
Data_type  - |0 = Empty||1 = Bool||2 = char||3 = short||4 = int||5 = float||6 = double|
Value_type - |0 = Empty||1 = nano||2 = micro||3 = mili||4 = centi||5 = deci||6 = deka||7 = hecto||8 = kilo||9 = mega|
Unit_type  - |0 = Empty||1 = celsius||2 = kilogram||3 = pascal||4 = meter_per_second||5 = meter|
Device_Serial - 8bit number
Device_Type - not specified 8bit number
Slave_Address  - 5 Bytes - Sensor receiving Address
Master_Address - 5 Bytes - Master receiving Address - MAX 5 Nodes = !!!Master Address HAVE TO BE IDENTICAL EXCEPT LAST BYTE!!!
Device_Description - string of max 19 character to describe Device
-----------------------------------------------------
$1/1/1/1;
#2;
>4/0/5/22/1/0xABABABABAB/0xC2C2C2C2C2/Test-Sens-1;
>4/0/5/33/1/0xCDCDCDCDCD/0xC2C2C2C2C3/Test-Sens-2;
```

Obr. F.1: Příklad konfiguračního souboru pro nastavení měření

G Příklad XML souboru z kontrolního měření

```
<?xml version="1.1"?>
<Measure time="21/04/24-14:21:14:124">
        <Header>
                <MasterName>DataConcentrator</MasterName>
                <BoardRev>1.1</BoardRev>
                <ProtocolVer>1.1</ProtocolVer>
                <CommSpeed>2Mbps</CommSpeed>
                <Modules>
                        <nRF24L01>1</nRF24L01>
                        <GPS>1</GPS>
                        <CLOUD_LOG>1</CLOUD_LOG>
                        <SD_LOG>1</SD_LOG>
                </Modules>
        </Header>
        <Measurement>
                <info>
                        <StartTime>"21/04/24-14:21:14:124"</StartTime>
                        <NumberOfNodes>4</NumberOfNodes>
                        <NodesDescription>
                                <NodeDesc>Test-Sens-1</NodeDesc>
                                <NodeDesc>Test-Sens-2</NodeDesc>
                                <NodeDesc>GPS coordinates</NodeDesc>
                                <NodeDesc>GPS Alt./Speed</NodeDesc>
                        </NodesDescription>
                </info>
                <node num="0">
                        <info>
                                <UID>0</UID>
                                <DataType>4</DataType>
                                <ValueType>0</ValueType>
                                <Unit>5</Unit>
                        </info>
                        <device>
                                <DeviceSerial>22</DeviceSerial>
                                <DeviceType>1</DeviceType>
                                <SlaveAddress>0xabababababab</SlaveAddress>
                                <MasterAddress>0xc2c2c2c2c2c2</MasterAddress>
                                <ProtocolVer>1.1</ProtocolVer>
                                <DeviceDesc>Test-Sens-1</DeviceDesc>
                        </device>
                        <data>
                                <log><value>0</value><timestamp>32005</timestamp></log>
                                <log><value>10</value><timestamp>61957</timestamp></log>
                                <log><value>20</value><timestamp>91909</timestamp></log>
                                <log><value>30</value><timestamp>121861</timestamp></log>
                                <log><value>40</value><timestamp>151813</timestamp></log>
                                <log><value>50</value><timestamp>181765</timestamp></log>
                                <log><value>60</value><timestamp>211973</timestamp></log>
                                <log><value>70</value><timestamp>241925</timestamp></log>
                                <log><value>80</value><timestamp>271877</timestamp></log>
                                <log><value>90</value><timestamp>301829</timestamp></log>
                                <log><value>0</value><timestamp>331781</timestamp></log>
                                <log><value>10</value><timestamp>361989</timestamp></log>
                                <log><value>20</value><timestamp>391941</timestamp></log>
                                <log><value>30</value><timestamp>421893</timestamp></log>
                                <log><value>40</value><timestamp>451845</timestamp></log>
                                <log><value>50</value><timestamp>481797</timestamp></log>
                                <log><value>60</value><timestamp>512005</timestamp></log>
                                <log><value>70</value><timestamp>541957</timestamp></log>
                                <log><value>80</value><timestamp>571909</timestamp></log>
                                <log><value>90</value><timestamp>601861</timestamp></log>
                                <log><value>0</value><timestamp>631813</timestamp></log>
                                <log><value>10</value><timestamp>661765</timestamp></log>
                                <log><value>20</value><timestamp>691973</timestamp></log>
                                <log><value>30</value><timestamp>721925</timestamp></log>
                                <log><value>40</value><timestamp>751877</timestamp></log>
                                <log><value>50</value><timestamp>781829</timestamp></log>
                                <log><value>60</value><timestamp>811781</timestamp></log>
                                <log><value>70</value><timestamp>841989</timestamp></log>
                                <log><value>80</value><timestamp>871941</timestamp></log>
                                <log><value>90</value><timestamp>901893</timestamp></log>
                                <log><value>0</value><timestamp>931845</timestamp></log>
                                <log><value>10</value><timestamp>961797</timestamp></log>
```

Obr. G.1: Příklad XML souboru z kontrolního měření - část 1

```
                        <log><value>20</value><timestamp>992005</timestamp></log>
                        <log><value>30</value><timestamp>1021957</timestamp></log>
                        <log><value>40</value><timestamp>1051909</timestamp></log>
                        <log><value>50</value><timestamp>1081861</timestamp></log>
                        <log><value>60</value><timestamp>1111813</timestamp></log>
                        <log><value>70</value><timestamp>1141765</timestamp></log>
                        <log><value>80</value><timestamp>1171973</timestamp></log>
<!--0-->
                </data>
        </node>
        <node num="1">
                <info>
                        <UID>1</UID>
                        <DataType>4</DataType>
                        <ValueType>0</ValueType>
                        <Unit>5</Unit>
                </info>
                <device>
                        <DeviceSerial>33</DeviceSerial>
                        <DeviceType>1</DeviceType>
                        <SlaveAddress>0xcdcdcdcdcd</SlaveAddress>
                        <MasterAddress>0xc2c2c2c2c3</MasterAddress>
                        <ProtocolVer>1.1</ProtocolVer>
                        <DeviceDesc>Test-Sens-2</DeviceDesc>
                </device>
                <data>
                        <log><value>0</value><timestamp>32773</timestamp></log>
                        <log><value>10</value><timestamp>62981</timestamp></log>
                        <log><value>20</value><timestamp>92933</timestamp></log>
                        <log><value>30</value><timestamp>122885</timestamp></log>
                        <log><value>40</value><timestamp>152837</timestamp></log>
                        <log><value>50</value><timestamp>182789</timestamp></log>
                        <log><value>60</value><timestamp>212997</timestamp></log>
                        <log><value>70</value><timestamp>242949</timestamp></log>
                        <log><value>80</value><timestamp>272901</timestamp></log>
                        <log><value>90</value><timestamp>302853</timestamp></log>
                        <log><value>100</value><timestamp>332805</timestamp></log>
                        <log><value>110</value><timestamp>362757</timestamp></log>
                        <log><value>120</value><timestamp>392965</timestamp></log>
                        <log><value>130</value><timestamp>422917</timestamp></log>
                        <log><value>140</value><timestamp>452869</timestamp></log>
                        <log><value>150</value><timestamp>482821</timestamp></log>
                        <log><value>160</value><timestamp>512773</timestamp></log>
                        <log><value>170</value><timestamp>542981</timestamp></log>
                        <log><value>180</value><timestamp>572933</timestamp></log>
                        <log><value>190</value><timestamp>602885</timestamp></log>
                        <log><value>0</value><timestamp>632837</timestamp></log>
                        <log><value>10</value><timestamp>662789</timestamp></log>
                        <log><value>20</value><timestamp>692997</timestamp></log>
                        <log><value>30</value><timestamp>722949</timestamp></log>
                        <log><value>40</value><timestamp>752901</timestamp></log>
                        <log><value>50</value><timestamp>782853</timestamp></log>
                        <log><value>60</value><timestamp>812805</timestamp></log>
                        <log><value>70</value><timestamp>842757</timestamp></log>
                        <log><value>80</value><timestamp>872965</timestamp></log>
                        <log><value>90</value><timestamp>902917</timestamp></log>
                        <log><value>100</value><timestamp>932869</timestamp></log>
                        <log><value>110</value><timestamp>962821</timestamp></log>
                        <log><value>120</value><timestamp>992773</timestamp></log>
                        <log><value>130</value><timestamp>1022981</timestamp></log>
                        <log><value>140</value><timestamp>1052933</timestamp></log>
                        <log><value>150</value><timestamp>1082885</timestamp></log>
                        <log><value>160</value><timestamp>1112837</timestamp></log>
                        <log><value>170</value><timestamp>1142789</timestamp></log>
                        <log><value>180</value><timestamp>1172997</timestamp></log>
<!--1-->
                </data>
        </node>
        <node num="2">
                <info>
                        <UID>2</UID>
                        <DataType>7</DataType>
                        <ValueType>0</ValueType>
```

Obr. G.2: Příklad XML souboru z kontrolního měření - část 2

```
                <Unit>0</Unit>
            </info>
            <device>
                <DeviceSerial>0</DeviceSerial>
                <DeviceType>0</DeviceType>
                <SlaveAddress>0x0</SlaveAddress>
                <MasterAddress>0x0</MasterAddress>
                <ProtocolVer>1.1</ProtocolVer>
                <DeviceDesc>GPS coordinates</DeviceDesc>
            </device>
            <data>
                <log><latitude>49.199169</latitude><longitude>16.602514</longitude><timestamp>45974</timestamp></log>
                <log><latitude>49.198967</latitude><longitude>16.602913</longitude><timestamp>75976</timestamp></log>
                <log><latitude>49.198437</latitude><longitude>16.603172</longitude><timestamp>105976</timestamp></log>
                <log><latitude>49.198261</latitude><longitude>16.603025</longitude><timestamp>135977</timestamp></log>
                <log><latitude>49.198124</latitude><longitude>16.602873</longitude><timestamp>165976</timestamp></log>
                <log><latitude>49.197964</latitude><longitude>16.602400</longitude><timestamp>195976</timestamp></log>
                <log><latitude>49.197731</latitude><longitude>16.602739</longitude><timestamp>225977</timestamp></log>
                <log><latitude>49.197224</latitude><longitude>16.602896</longitude><timestamp>255977</timestamp></log>
                <log><latitude>49.196999</latitude><longitude>16.603188</longitude><timestamp>285974</timestamp></log>
                <log><latitude>49.196659</latitude><longitude>16.603395</longitude><timestamp>315976</timestamp></log>
                <log><latitude>49.196274</latitude><longitude>16.603603</longitude><timestamp>345977</timestamp></log>
                <log><latitude>49.195923</latitude><longitude>16.603901</longitude><timestamp>375976</timestamp></log>
                <log><latitude>49.195518</latitude><longitude>16.604177</longitude><timestamp>405976</timestamp></log>
                <log><latitude>49.195068</latitude><longitude>16.604090</longitude><timestamp>435976</timestamp></log>
                <log><latitude>49.195004</latitude><longitude>16.604071</longitude><timestamp>465978</timestamp></log>
                <log><latitude>49.194717</latitude><longitude>16.604198</longitude><timestamp>495977</timestamp></log>
                <log><latitude>49.194283</latitude><longitude>16.604324</longitude><timestamp>525977</timestamp></log>
                <log><latitude>49.193851</latitude><longitude>16.604534</longitude><timestamp>555978</timestamp></log>
                <log><latitude>49.193474</latitude><longitude>16.604639</longitude><timestamp>585976</timestamp></log>
                <log><latitude>49.193401</latitude><longitude>16.604609</longitude><timestamp>615975</timestamp></log>
                <log><latitude>49.193638</latitude><longitude>16.604397</longitude><timestamp>645977</timestamp></log>
                <log><latitude>49.193974</latitude><longitude>16.604235</longitude><timestamp>675978</timestamp></log>
                <log><latitude>49.194393</latitude><longitude>16.603983</longitude><timestamp>705977</timestamp></log>
                <log><latitude>49.194843</latitude><longitude>16.603815</longitude><timestamp>735977</timestamp></log>
                <log><latitude>49.195000</latitude><longitude>16.603634</longitude><timestamp>765978</timestamp></log>
                <log><latitude>49.195145</latitude><longitude>16.603292</longitude><timestamp>795978</timestamp></log>
                <log><latitude>49.195312</latitude><longitude>16.602745</longitude><timestamp>825977</timestamp></log>
                <log><latitude>49.195625</latitude><longitude>16.602558</longitude><timestamp>855976</timestamp></log>
                <log><latitude>49.196117</latitude><longitude>16.602369</longitude><timestamp>885978</timestamp></log>
                <log><latitude>49.196350</latitude><longitude>16.601883</longitude><timestamp>915978</timestamp></log>
                <log><latitude>49.196747</latitude><longitude>16.601446</longitude><timestamp>945975</timestamp></log>
                <log><latitude>49.197113</latitude><longitude>16.601294</longitude><timestamp>975977</timestamp></log>
                <log><latitude>49.197411</latitude><longitude>16.601650</longitude><timestamp>1005976</timestamp></log>
                <log><latitude>49.197632</latitude><longitude>16.602108</longitude><timestamp>1035976</timestamp></log>
                <log><latitude>49.198109</latitude><longitude>16.602512</longitude><timestamp>1065977</timestamp></log>
                <log><latitude>49.198307</latitude><longitude>16.602985</longitude><timestamp>1095977</timestamp></log>
                <log><latitude>49.198662</latitude><longitude>16.603317</longitude><timestamp>1125978</timestamp></log>
                <log><latitude>49.198887</latitude><longitude>16.603203</longitude><timestamp>1155978</timestamp></log>
<!--2-->
            </data>
        </node>
        <node num="3">
            <info>
                <UID>3</UID>
                <DataType>5</DataType>
                <ValueType>0</ValueType>
                <Unit>0</Unit>
            </info>
            <device>
                <DeviceSerial>0</DeviceSerial>
                <DeviceType>0</DeviceType>
                <SlaveAddress>0x0</SlaveAddress>
                <MasterAddress>0x0</MasterAddress>
                <ProtocolVer>1.1</ProtocolVer>
                <DeviceDesc>GPS Alt./Speed</DeviceDesc>
            </device>
            <data>
                <log><value1>258.600006</value1><value2>12.228500</value2><timestamp>45974</timestamp></log>
                <log><value1>250.399994</value1><value2>1.961000</value2><timestamp>75976</timestamp></log>
                <log><value1>281.399994</value1><value2>8.528500</value2><timestamp>105976</timestamp></log>
                <log><value1>276.899994</value1><value2>2.886000</value2><timestamp>135977</timestamp></log>
                <log><value1>278.399994</value1><value2>9.009500</value2><timestamp>165976</timestamp></log>
```

Obr. G.3: Příklad XML souboru z kontrolního měření - část 3

```
                        <log><value1>274.100006</value1><value2>7.770000</value2><timestamp>195976</timestamp></log>
                        <log><value1>260.200012</value1><value2>9.453500</value2><timestamp>225977</timestamp></log>
                        <log><value1>256.799988</value1><value2>6.234500</value2><timestamp>255977</timestamp></log>
                        <log><value1>246.699997</value1><value2>8.158500</value2><timestamp>285974</timestamp></log>
                        <log><value1>232.899994</value1><value2>9.731000</value2><timestamp>315976</timestamp></log>
                        <log><value1>228.500000</value1><value2>8.972500</value2><timestamp>345977</timestamp></log>
                        <log><value1>229.899994</value1><value2>13.301500</value2><timestamp>375976</timestamp></log>
                        <log><value1>231.699997</value1><value2>2.201500</value2><timestamp>405976</timestamp></log>
                        <log><value1>224.899994</value1><value2>8.010500</value2><timestamp>435976</timestamp></log>
                        <log><value1>226.199997</value1><value2>0.573500</value2><timestamp>465978</timestamp></log>
                        <log><value1>228.600006</value1><value2>6.234500</value2><timestamp>495977</timestamp></log>
                        <log><value1>228.199997</value1><value2>4.828500</value2><timestamp>525977</timestamp></log>
                        <log><value1>228.399994</value1><value2>2.053500</value2><timestamp>555978</timestamp></log>
                        <log><value1>226.000000</value1><value2>6.438000</value2><timestamp>585976</timestamp></log>
                        <log><value1>227.699997</value1><value2>3.422500</value2><timestamp>615975</timestamp></log>
                        <log><value1>227.600006</value1><value2>3.089500</value2><timestamp>645977</timestamp></log>
                        <log><value1>228.899994</value1><value2>2.849000</value2><timestamp>675978</timestamp></log>
                        <log><value1>226.399994</value1><value2>4.347500</value2><timestamp>705977</timestamp></log>
                        <log><value1>222.100006</value1><value2>6.789500</value2><timestamp>735977</timestamp></log>
                        <log><value1>223.600006</value1><value2>6.105000</value2><timestamp>765978</timestamp></log>
                        <log><value1>224.300003</value1><value2>5.587000</value2><timestamp>795978</timestamp></log>
                        <log><value1>219.800003</value1><value2>6.475000</value2><timestamp>825977</timestamp></log>
                        <log><value1>218.899994</value1><value2>7.326000</value2><timestamp>855976</timestamp></log>
                        <log><value1>216.800003</value1><value2>11.211000</value2><timestamp>885978</timestamp></log>
                        <log><value1>211.899994</value1><value2>8.158500</value2><timestamp>915978</timestamp></log>
                        <log><value1>211.199997</value1><value2>10.415500</value2><timestamp>945975</timestamp></log>
                        <log><value1>209.699997</value1><value2>4.625000</value2><timestamp>975977</timestamp></log>
                        <log><value1>208.000000</value1><value2>6.586000</value2><timestamp>1005976</timestamp></log>
                        <log><value1>207.300003</value1><value2>6.401000</value2><timestamp>1035976</timestamp></log>
                        <log><value1>208.000000</value1><value2>9.083500</value2><timestamp>1065977</timestamp></log>
                        <log><value1>206.100006</value1><value2>10.341500</value2><timestamp>1095977</timestamp></log>
                        <log><value1>211.699997</value1><value2>5.920000</value2><timestamp>1125978</timestamp></log>
                        <log><value1>212.699997</value1><value2>7.955000</value2><timestamp>1155978</timestamp></log>
<!--3-->
                    </data>
                </node>
<!--N-->
        </Measurement>
</Measure>
```

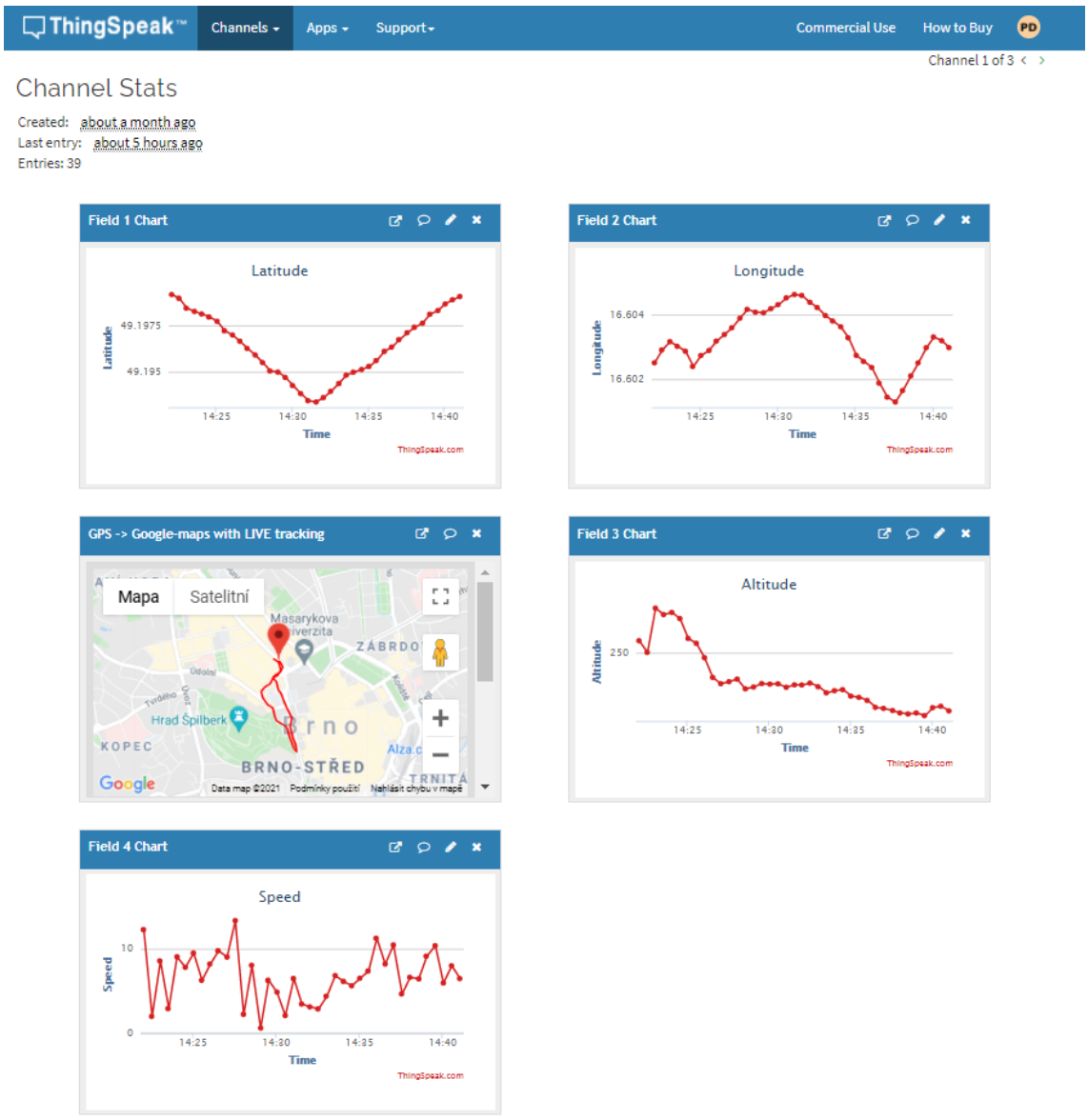
Obr. G.4: Příklad XML souboru z kontrolního měření - část 4

H Testovací skript v prostředí ThingSpeak

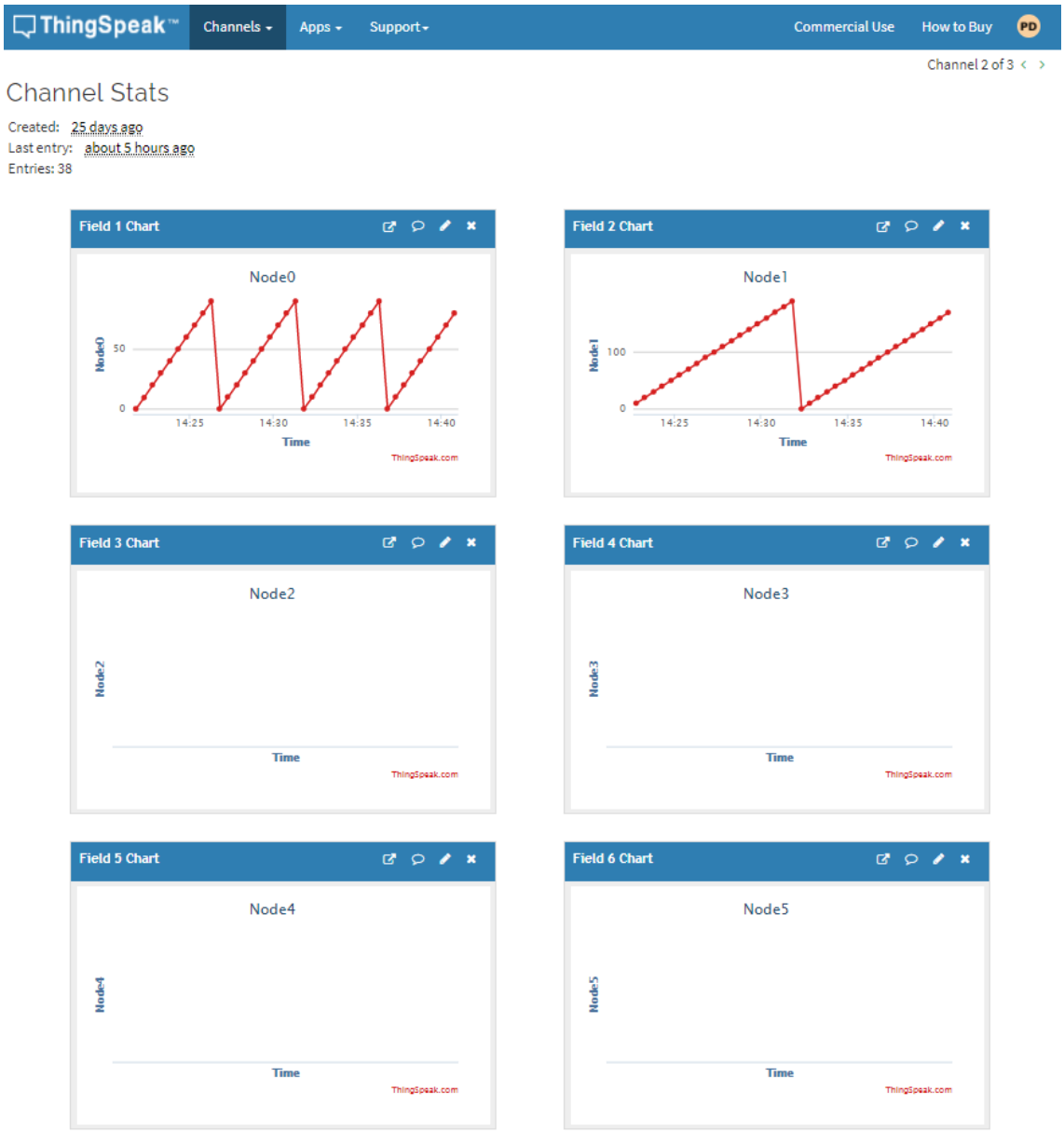
Výpis H.1: Příklad skriptu pro prostředí Matlab v rámci platformy ThingSpeak

```
%% Matlab Test script
% Channel ID to read data from / Channel Read API Key
rChID = XXXXXXX;
readAPIKey = 'XXXXXXXXXXXXXXXX';
[data,timestamps,channelInfo] = thingSpeakRead(rChID,...
                    'Fields',[1,2],...
                    'NumPoints', 4,...
                    'ReadKey',readAPIKey);
data(:,1) %Node0
data(:,2) %Node1
if isnan(data(end, 1))
    avgNode0 = mean(data(1:end-1,1));
else
    avgNode0 = mean(data(:,1));
end
if isnan(data(end, 2))
   avgNode1 = mean(data(1:end-1,2));
else
    avgNode1 = mean(data(:,2));
end
display(avgNode0,'Average␣of␣Node0');
display(avgNode1,'Average␣of␣Node1');
% Channel ID to write data to /  Channel Write API Key
wChID = XXXXXXX;
writeAPIKey = 'XXXXXXXXXXXXXXXX';
thingSpeakWrite(writeChannelID,[avgNode0, avgNode1],...
            'Fields',[1,2],...
            'WriteKey',writeAPIKey);
```

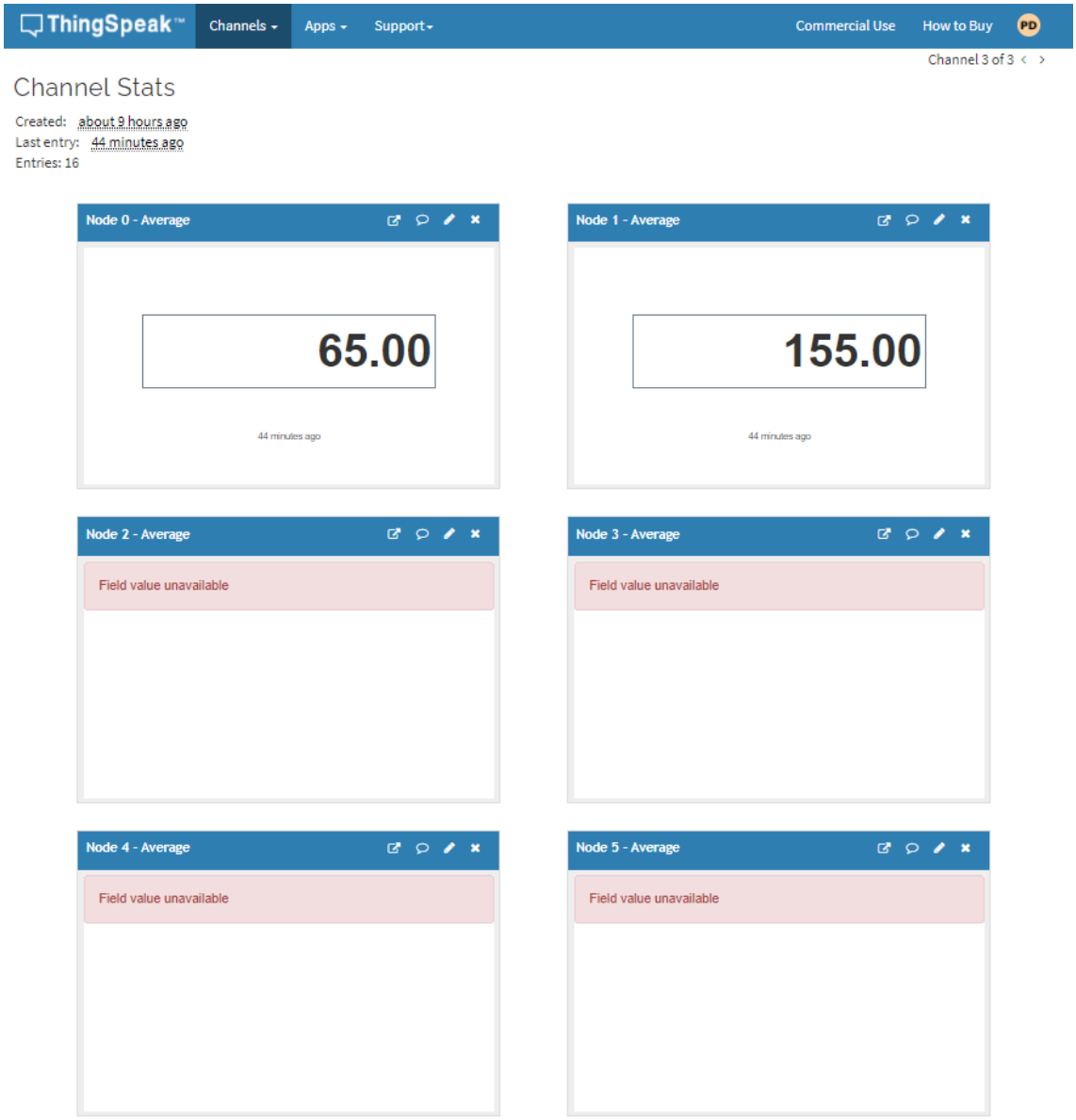
I Měřicí kanály v prostředí ThingSpeak



Obr. I.1: Uživatelské zobrazení kanálu GPS v prostředí ThingSpeak



Obr. I.2: Uživatelské zobrazení kanálu měřicích uzlů v prostředí ThingSpeak



Obr. I.3: Uživatelské zobrazení zpracovatelského kanálu v prostředí ThingSpeak

J Obsah přílohy na přiloženém CD

- / ..... Kořenový adresář přiloženého souboru
  - Eagle_Data_Concentrator ..... Elektrická dokumentace Datového koncentrátoru
    - DataConcentratorFabricationFiles ..... Dokumentace pro výrobu desky
    - DataConterator ..... Elektrický návrh Datového koncentrátoru
      - Data_Koncentrator_DPS.pdf
      - Data_Koncentrator_Elektrické_Shéma.pdf
      - Data_Koncentrator.brd
      - Data_Koncentrator.pro
      - Data_Koncentrator.sch
      - Eagle.epf
  - VScode_PIO_HW_TEST ..... Zkušební prog. vybavení Koncentrátoru
    - lib
    - src
      - DS3231SN_main.cpp
      - INPUTS_OUTPUTS_main.cpp
      - microSD_MMC_main.cpp
      - microSD_SPI_main.cpp
      - nRF24L01_main.cpp
      - SIM808_main.cpp
  - VScode_PIO_DataConcentrator ..... Kompletní prog. vybavení Koncentrátoru
    - lib
    - src
      - main.cpp
    - platformio.ini
  - VScode_PIO_Pseudo_Sensor ..... Testovací prog. vybavení pro pseudo snímač
    - lib
    - src
      - DC_TEST_Pseudo_Sensor.cpp
  - ThingSpeak ..... Softwarove komponenty a skript pro ThingSpeak
    - ThingSpeak_GoogleMaps_Plugin
      - ThingSpeak_GoogleMaps_Plugin_CSS.css
      - ThingSpeak_GoogleMaps_Plugin_HTML.html
      - ThingSpeak_GoogleMaps_Plugin_JavaScript.js
    - ThingSpeak_Matlab_Script_AverageNodes.m
  - Kontrolni_mereni ..... Soubory pro konfiguraci měření a jeho výsledky
    - 21_04_24
      - 14_21_14.XML
    - SETTINGS
      - config.txt
    - ThingSpeak_GPS_Page.jpg
    - ThingSpeak_Measure_Analytics_Page.jpg
    - ThingSpeak_Measure_Nodes_Page.jpg
  - obrazky ..... Použité obrázky
    - DataConcentrator_Predni_Strana.jpg

- DataConcentrator_Zadni_Strana.jpg
- Render_DataConcentrator_Predni_Strana.jpg
- Render_DataConcentrator_Zadni_Strana.jpg

text ..... Zdrojové textové soubory
- hardware.tex
- literatura.bib
- literatura.tex
- prilohy.tex
- reseni.tex
- software.tex
- thingspeak.tex
- uvod.tex
- vysledky.tex
- zaver.tex
- zkratky.tex

nastaveni.tex ..... Soubor pro nastavení semestrální práce

sablona-prace.tex ..... Hlavní soubor pro sazbu semestrální práce

thesis.sty ..... Balíček pro sazbu semestrální prací